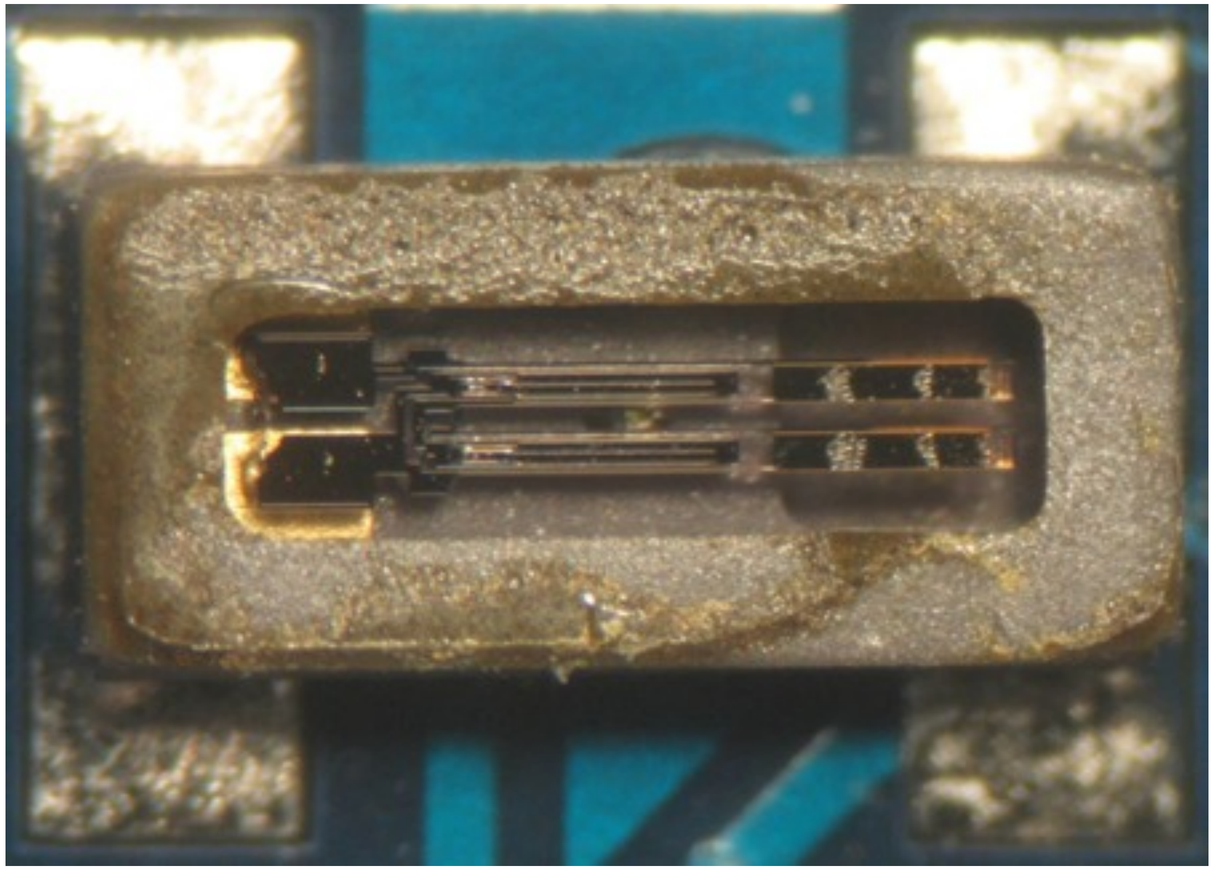
EECS 373

Design of Microprocessor-Based Systems



Thomas Schmid

University of Michigan

Lecture 8: Timers: count, compare, capture, PWM

September 28, 2010

http://home.netcom.com/~swansont/science.html

Minute Quiz...

Announcements

- Homework 1 posted on website
  - Due date October 7th

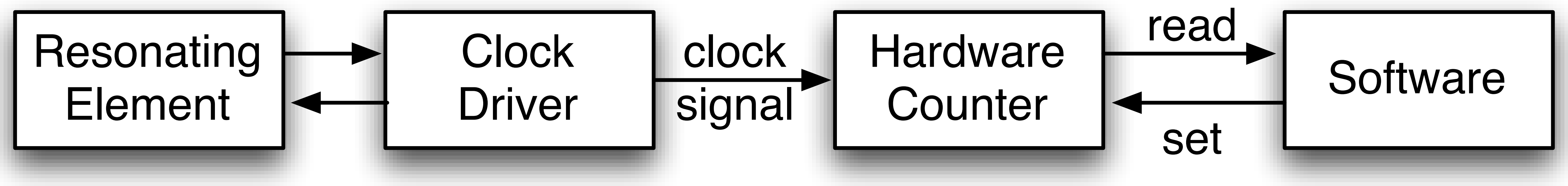
Where do we use time in an embedded system?

Why do we need accurate time?

- Scheduling of computation
  - Scheduler in operating systems
  - Real time operating systems
- Signal sampling
  - Audio sampling at 44.1 kHz
  - Sampling CCD at 30 fps
- Signal generation
  - 120 Hz TV refresh rate
  - Pulse Width Modulated (PWM) signals
- Communication
  - Media Access Control (MAC) protocols
  - Modulation
- Navigation
  - GPS
- Coordination

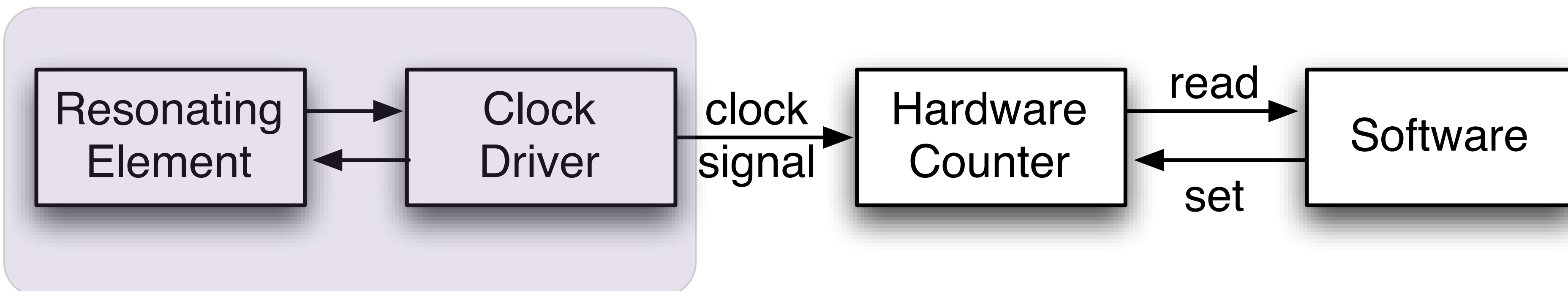
ABB Motion Control

Time in Embedded Systems



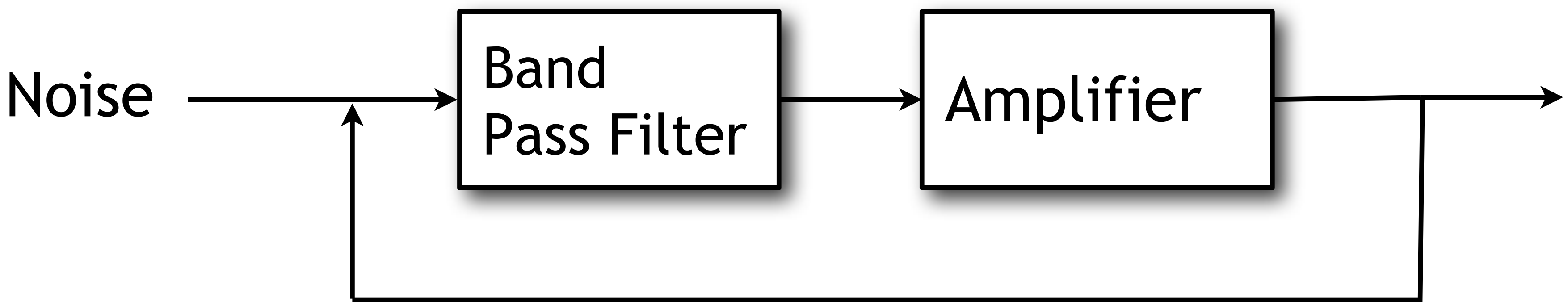
- Time is kept by a hardware counter, updated by a clock signal
- The clock signal increments the counter every *1/f* seconds (**resolution**)
- The counter reads $c(t) = \lfloor f \cdot t \rfloor \mod 2^n$
  - n: size of counter
- Smallest increment at which software can reader counter: **precision**
- How close is timer to UTC: **accuracy**

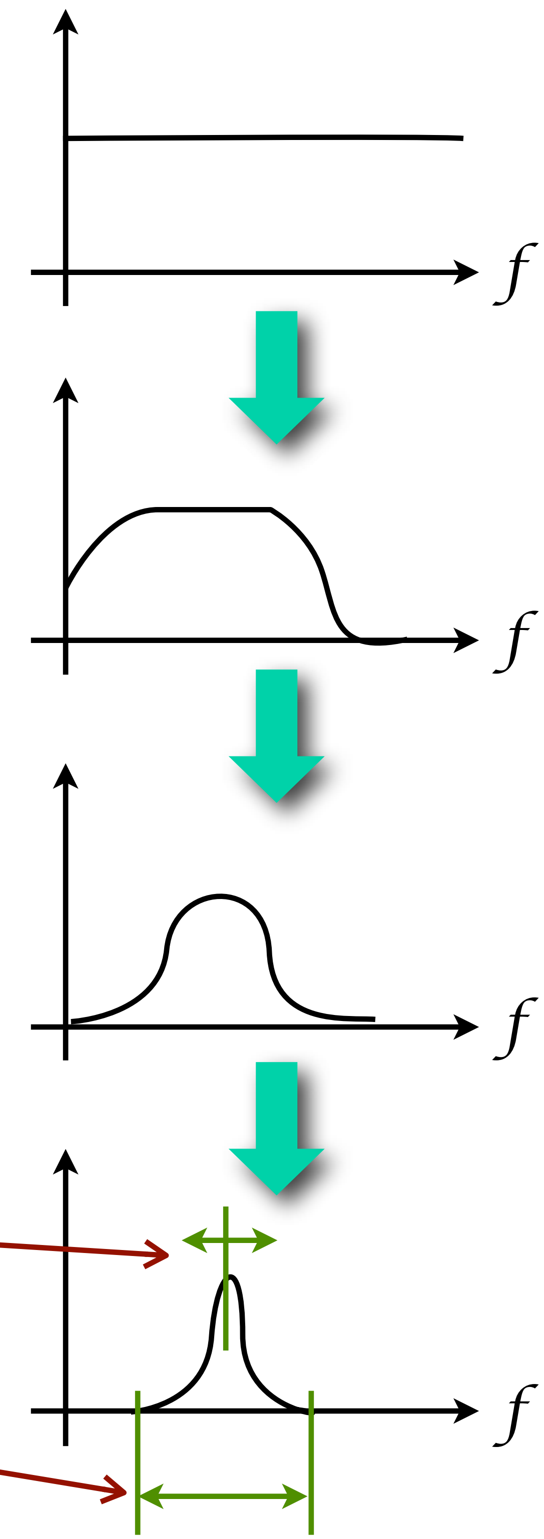
Resonator Technology



- LC/RC Circuits
- Inverter Ring
- Quartz Crystal
- MEMS Resonators
- Atomic Clock: Hydrogen Maser
- Others: Cesium, Rubidium, Ceramic, Bulk Acoustic Wave, Surface Acoustic Wave, Opto-electronic Oscillator, etc

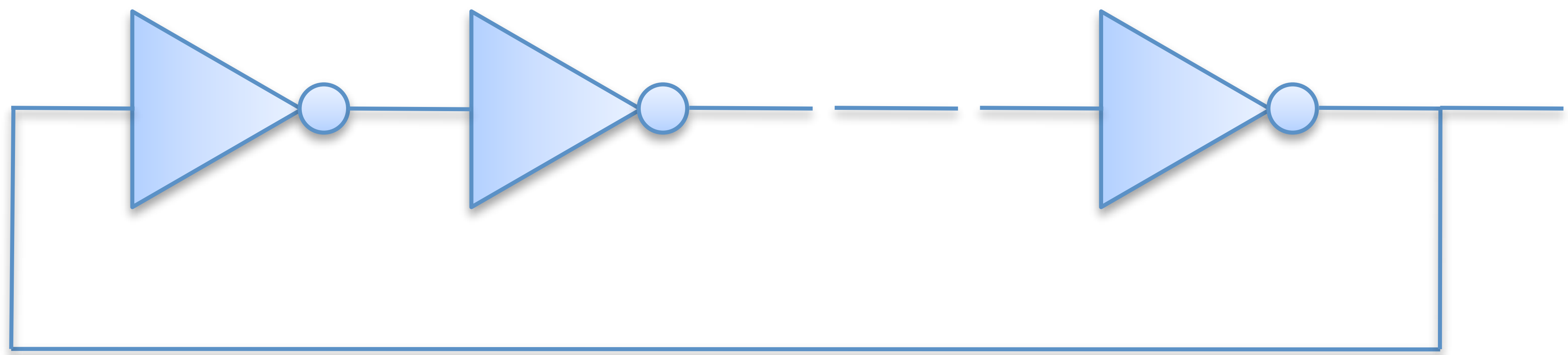
Resonator As Filter





- Barkhausen Criteria:
  - For a positive feedback system, oscillation will occur when loop gain (product of forward gain and feedback gain) has zero phase shift and a magnitude greater than unity.
- Performance Metrics
  - Quality or Q factor: measure of energy loss within resonating structure.
  - Frequency Stability: How much the center of the peak moves (longer term).
  - Phase Noise: Energy around the peak (short term).

Inverter Ring



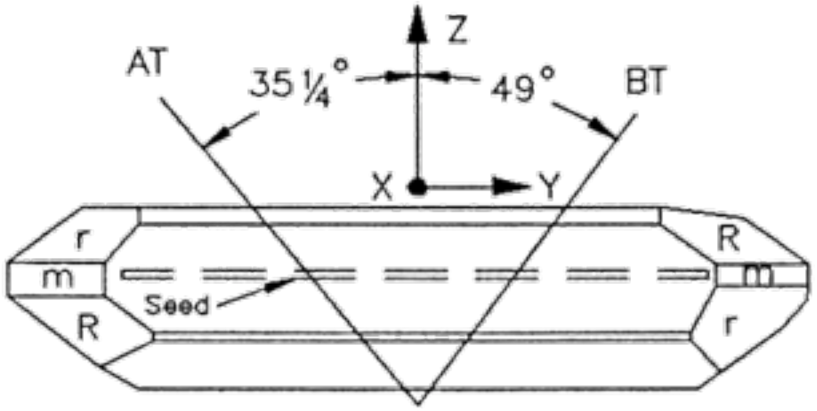
- An odd number of inverters arranged in a ring produce a frequency

$$f(T) = 1/2N \cdot t_{pd}(T)$$

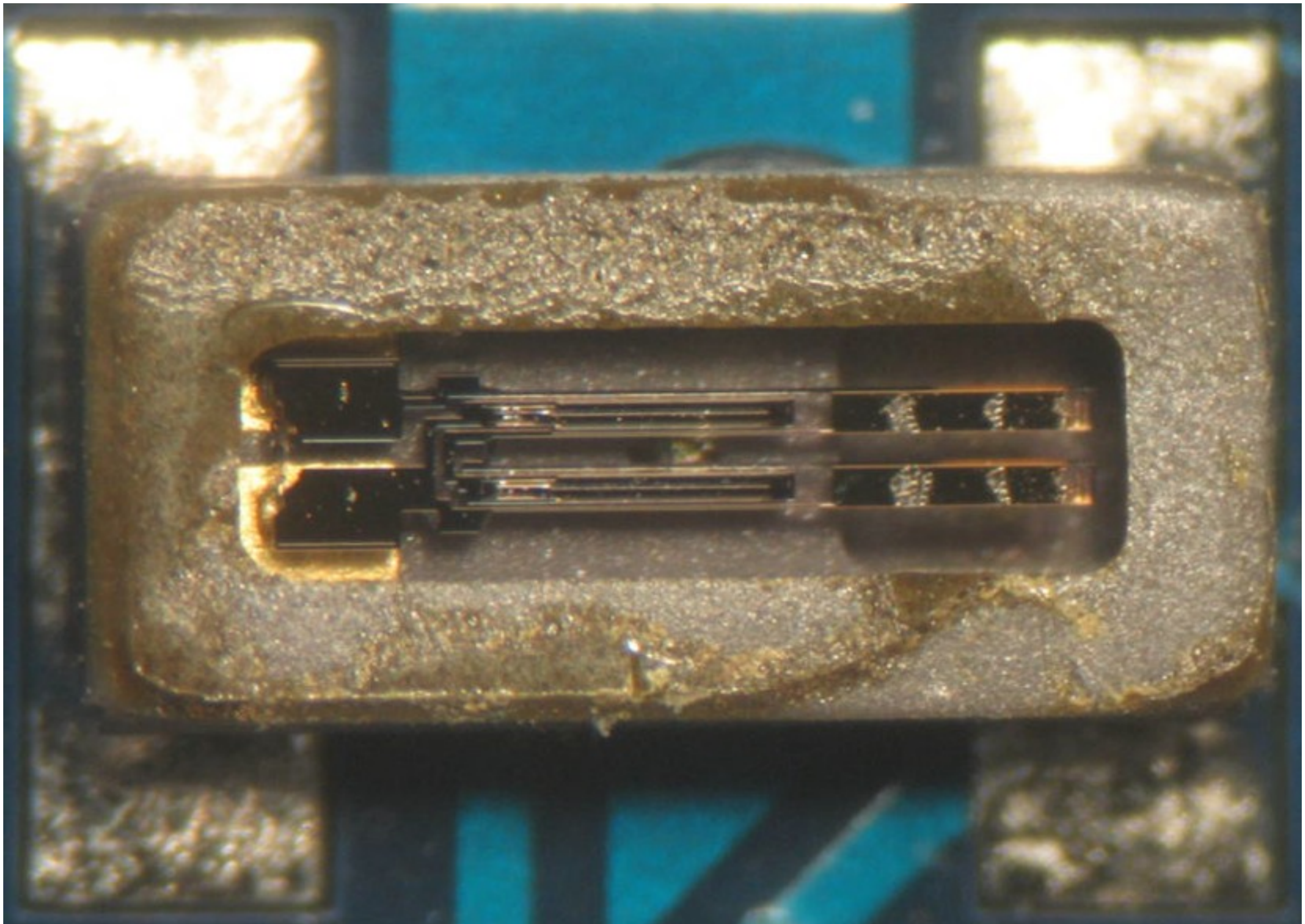
- Inverter propagation delay has strong temperature dependence, leading to frequency drift.
- Advantages:
  - Very high frequencies possible (tpd < 10ps for 90nm technology), high integration, almost zero cost when building a large chip, nearly arbitrary frequency choice.
- Disadvantages:
  - Very low Q-factor, very low stability ≈ 10^5 ppm (affected by temperature and voltage), very high phase noise.

Quartz Crystal

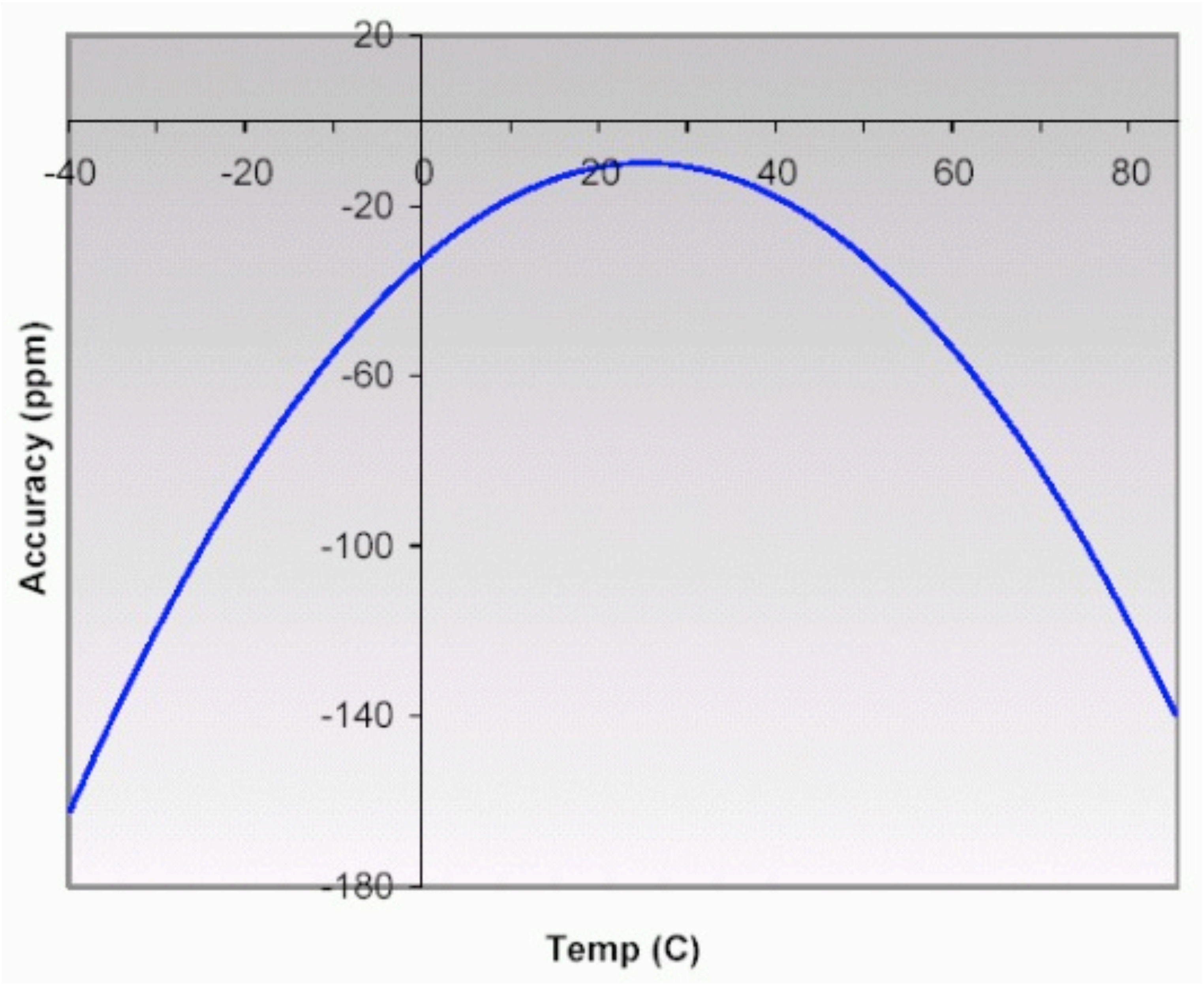
- Chemically, quartz is Silicon Dioxide and displays the Piezoelectric effect.
  - When a crystal of quartz is properly cut and mounted, it can be made to bend in an electric field.
  - When the field is removed, the quartz will generate an electric field as it returns to its previous shape.
- The resonance frequency of a quartz crystal depends on its length, thickness and angle of cut with respect to the crystallographic axes.
- Some angles have high immunity to temperature variations.
- Advantages:
  - Very high Q factor $\approx 10^6$, high stability $< 10^2$ ppm, low phase noise.
- Disadvantage:
  - Expensive, precision engineering, not all frequencies possible with all cuts.



Tuning Fork Crystal (magnified view)



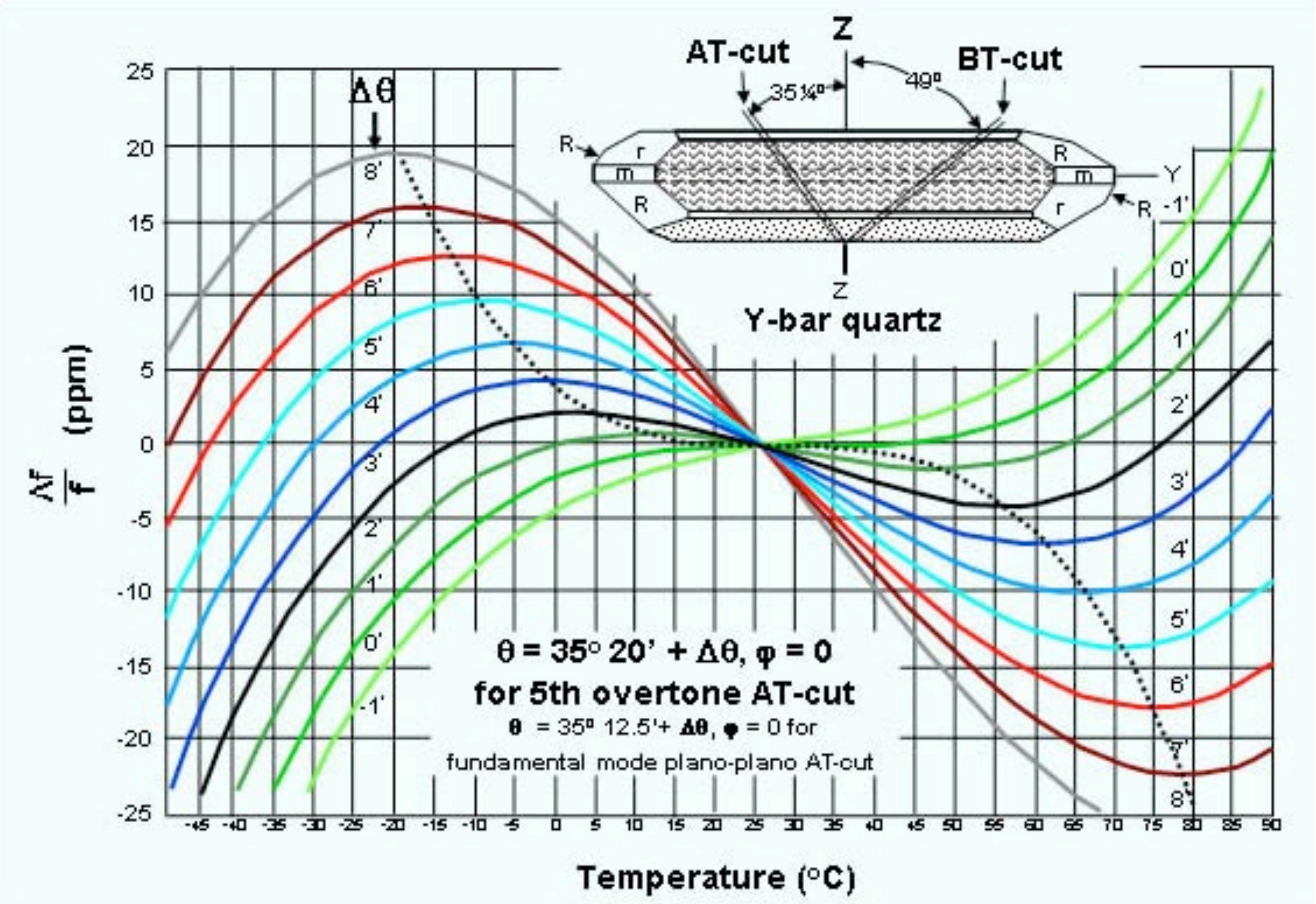
Temperature Dependence of Tuning Fork
Most common 32 kHz clock source



- Quadratic curve with zero ppm set at room temperature.

[Maxim-IC]

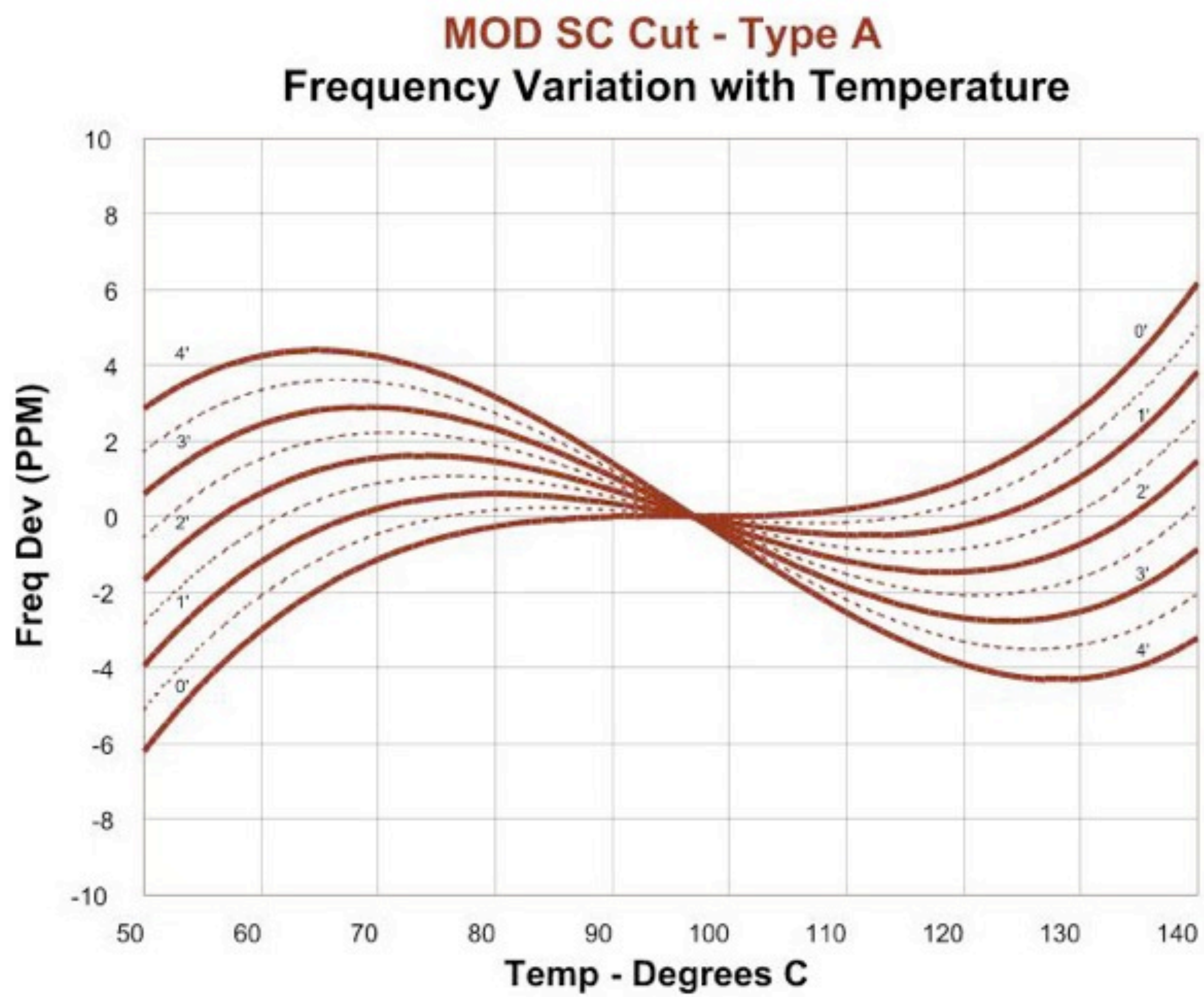
Temperature Dependence of AT-cut Quartz
Most common clock source >400 kHz



- Follows a cubic curve with parameters highly dependent on the angle of cut.

Z-Cut, SC-Cut, and many others...

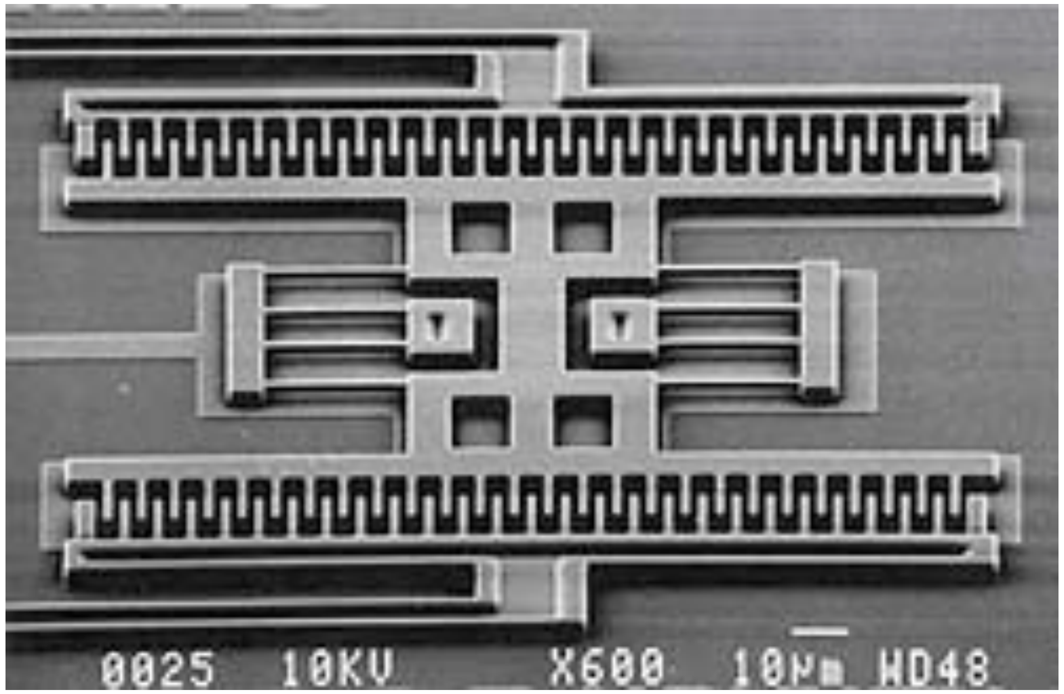
- SC-Cut is a doubly-rotated
- Can be excited in two modes at the same time!



[Bliley Technologies Inc]

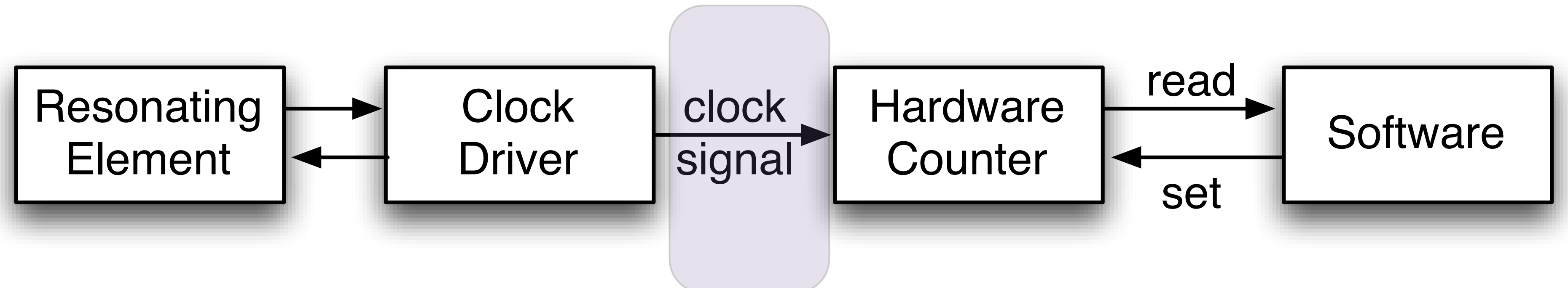
MEMS Resonator

- Micromachined structure designed for a specific resonant frequency - a tiny tuning fork.
- Exploiting silicon fabrication processes to precision engineer resonant structures at very low cost.
- Advantages: high Q-factor: 10^3-10^4, arbitrary frequency choice, large design space for future optimizations.
- Disadvantage: susceptible to temperature variations, high phase noise.

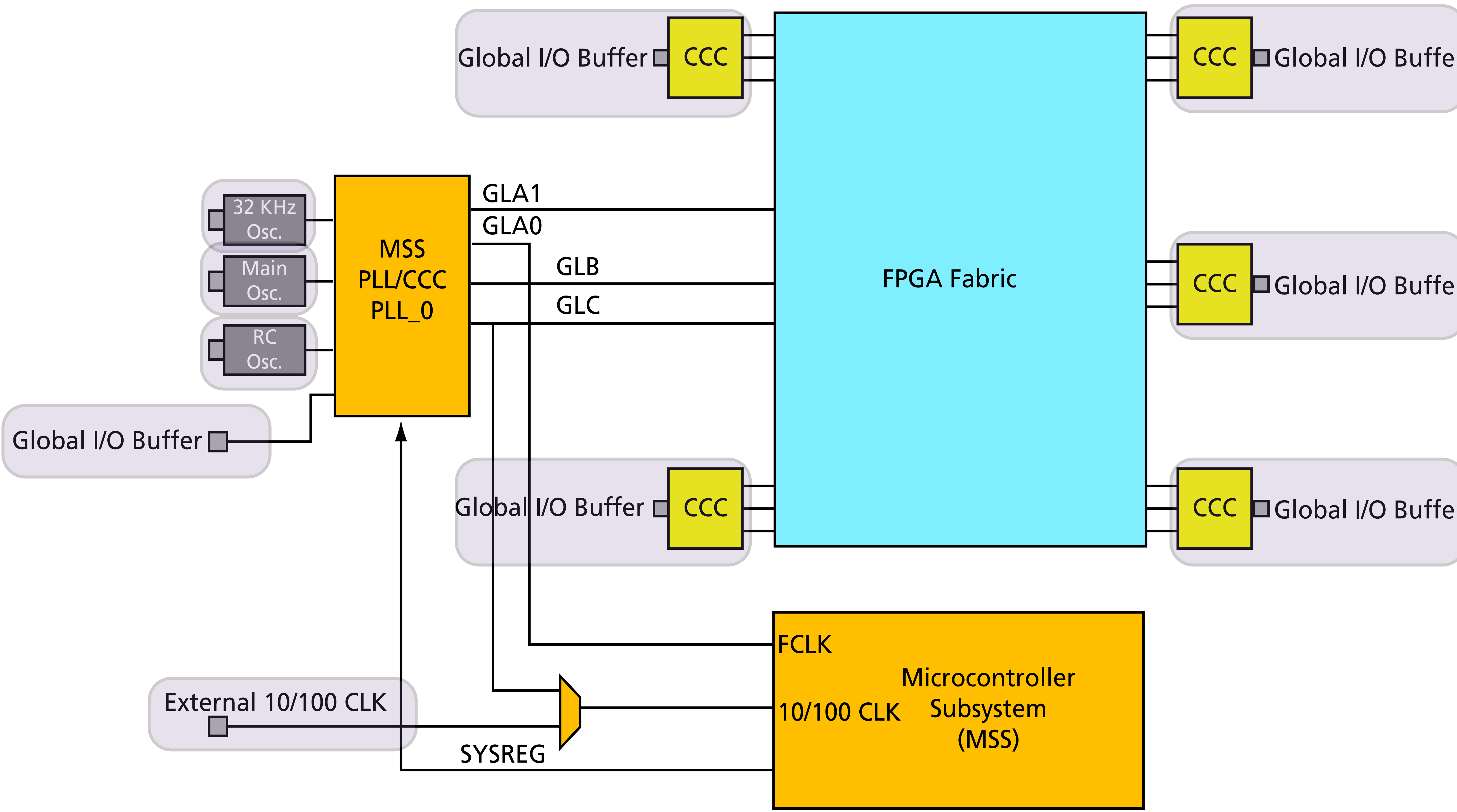


Clock Signals

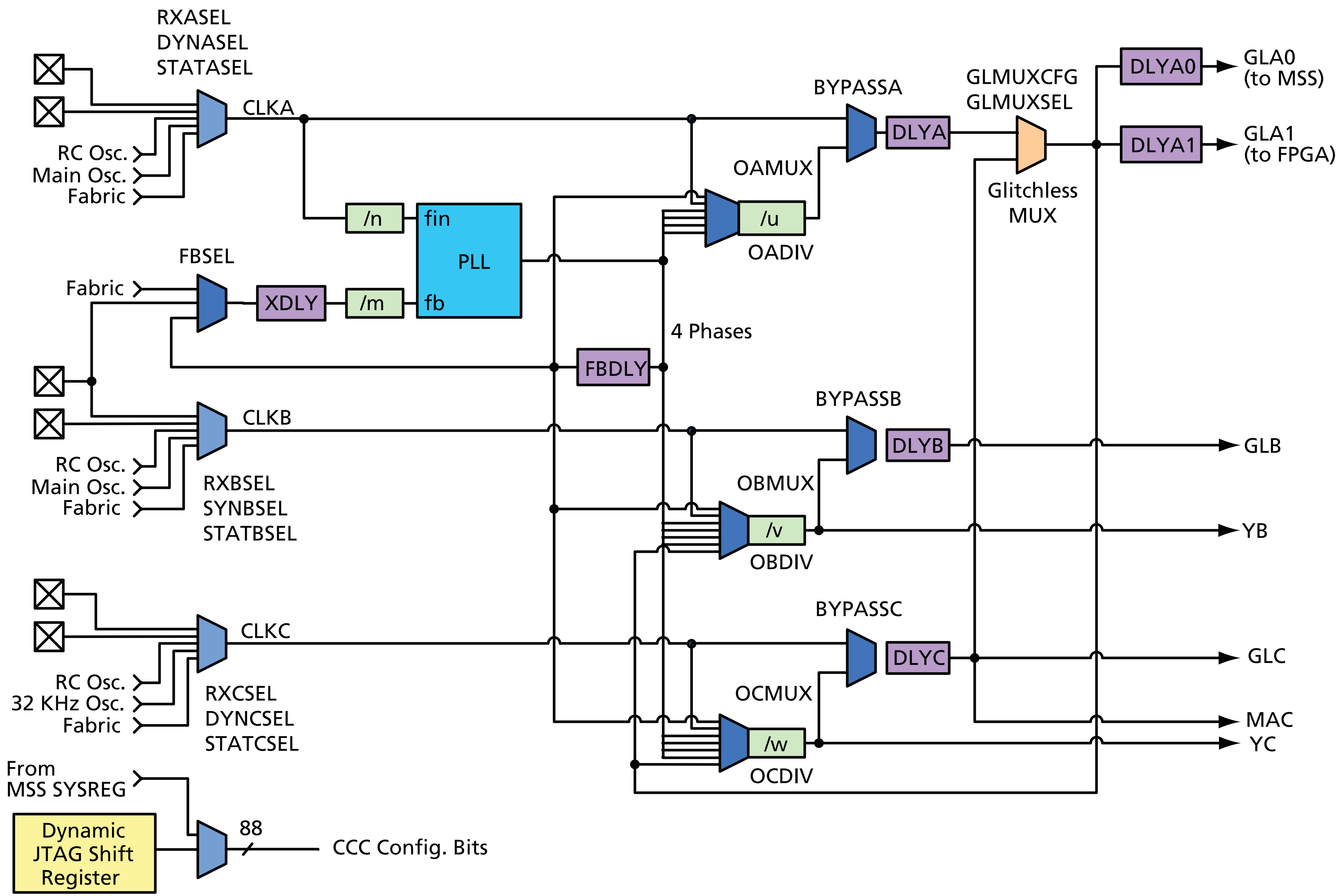
- How do we distribute and generate different clock signals?



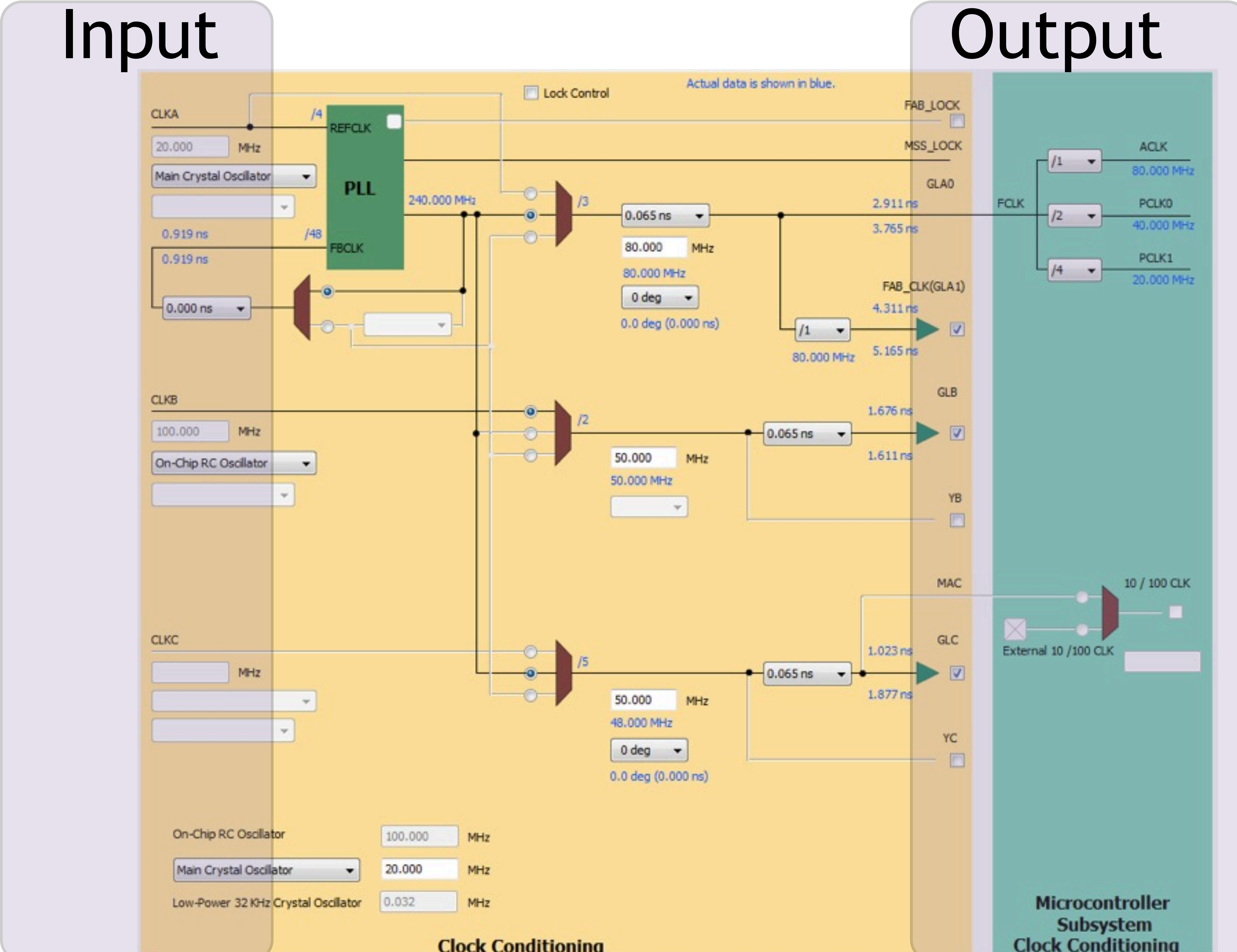
SmartFusion Clock Hierarchy



SmartFusion MSS Clock Conditioning Circuit

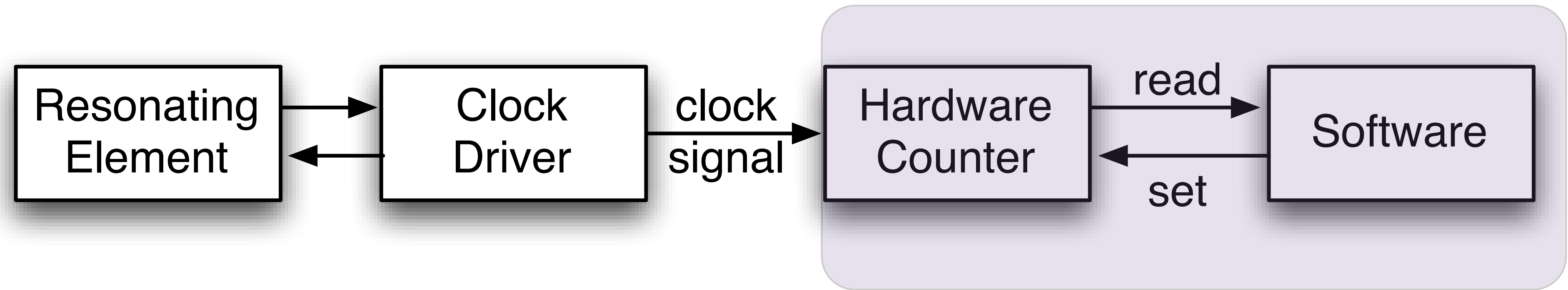


MSS Clock(s) Configurator



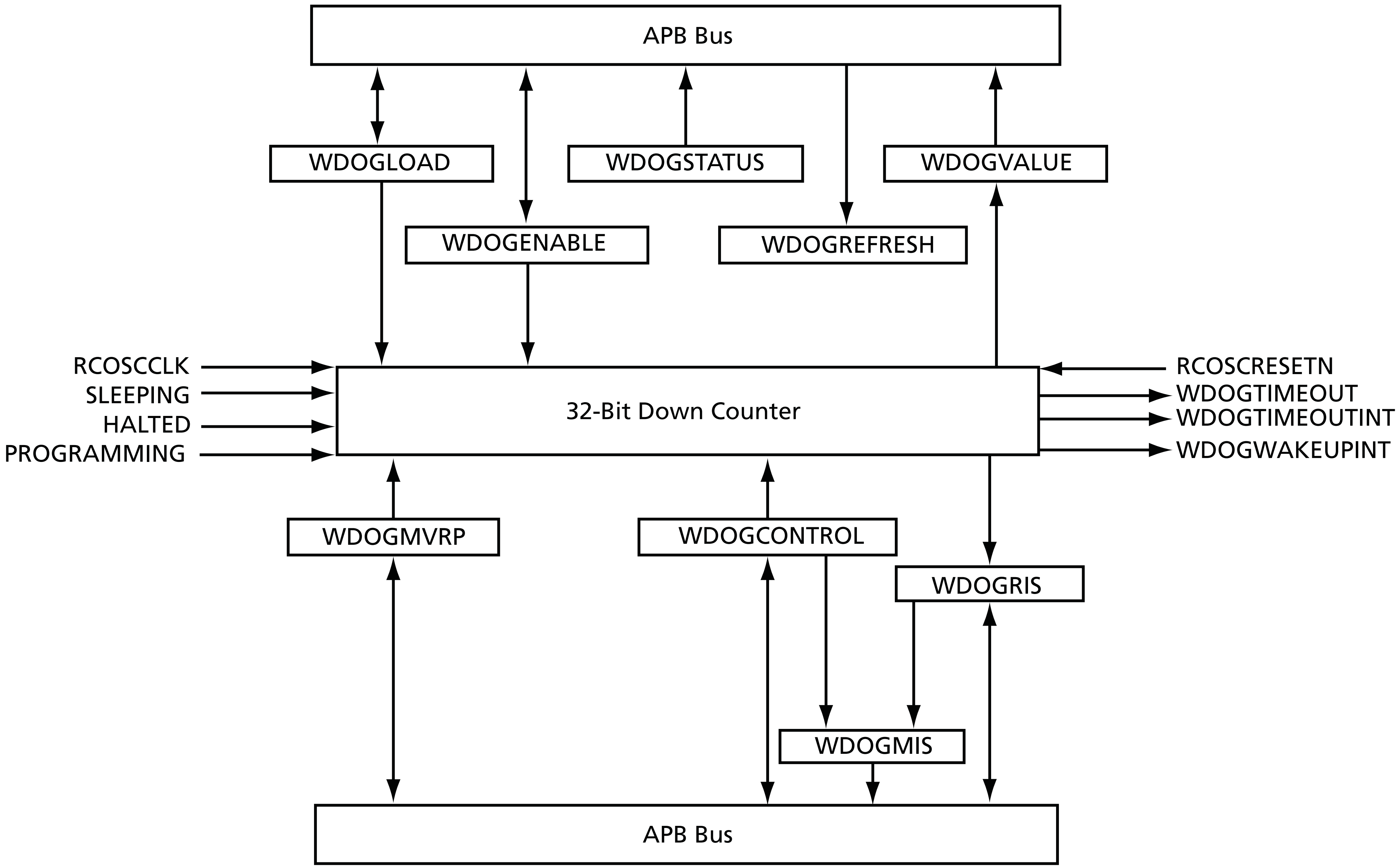
Timers, Capture, Compare, PWM

- How do we keep time?



Timers on the SmartFusion

- Watchdog Timer
  - 32-bit down counter
  - Either reset system or NMI Interrupt if it reaches 0!

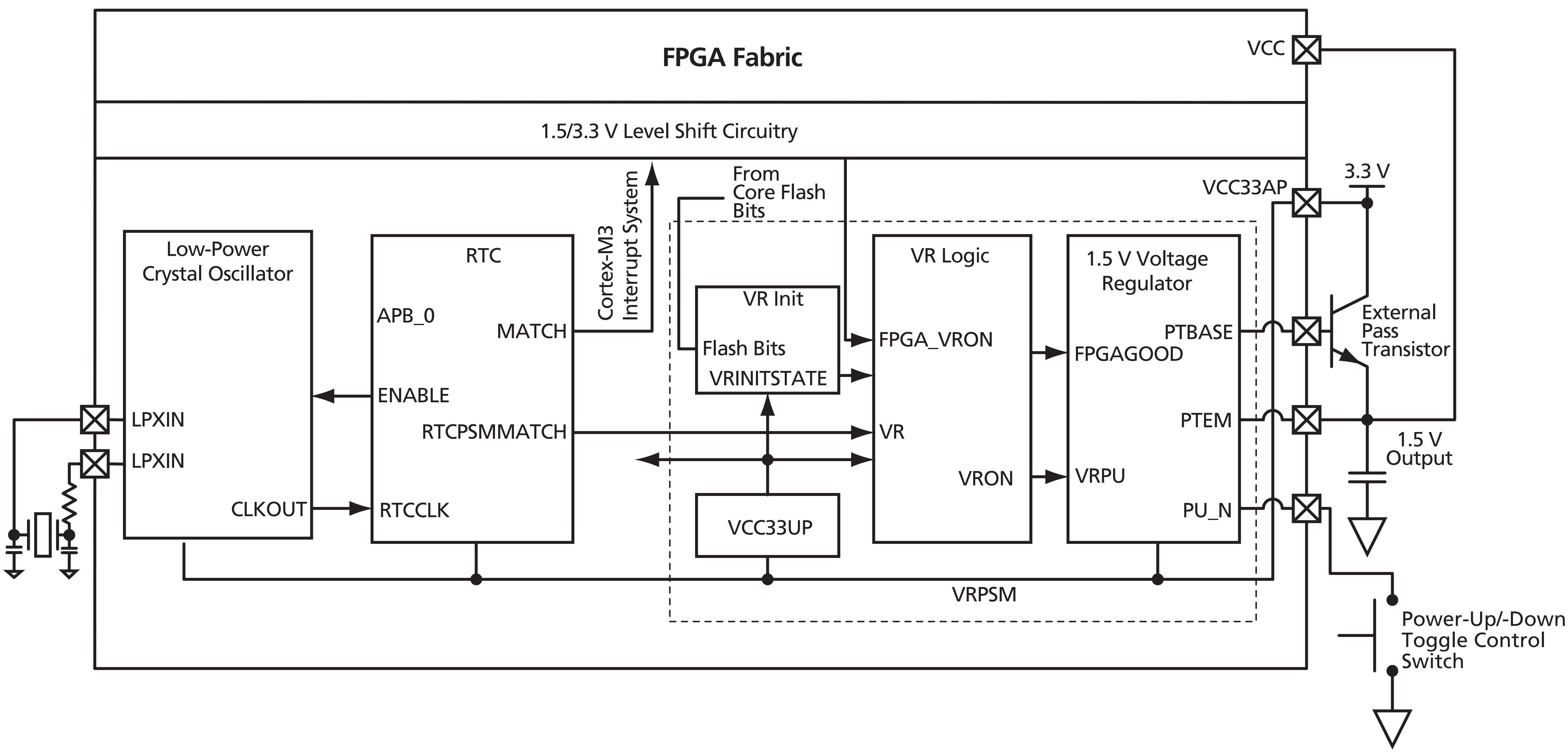


Timers on the SmartFusion (2)

- SysTick Timer
  - ARM requires every Cortex-M3 to have this timer
  - Essentially a 24-bit down-counter to generate system ticks
  - Has its own interrupt
  - Clocked by FCLK with optional programmable divider
- See Actel SmartFusion MSS User Guide for register definitions

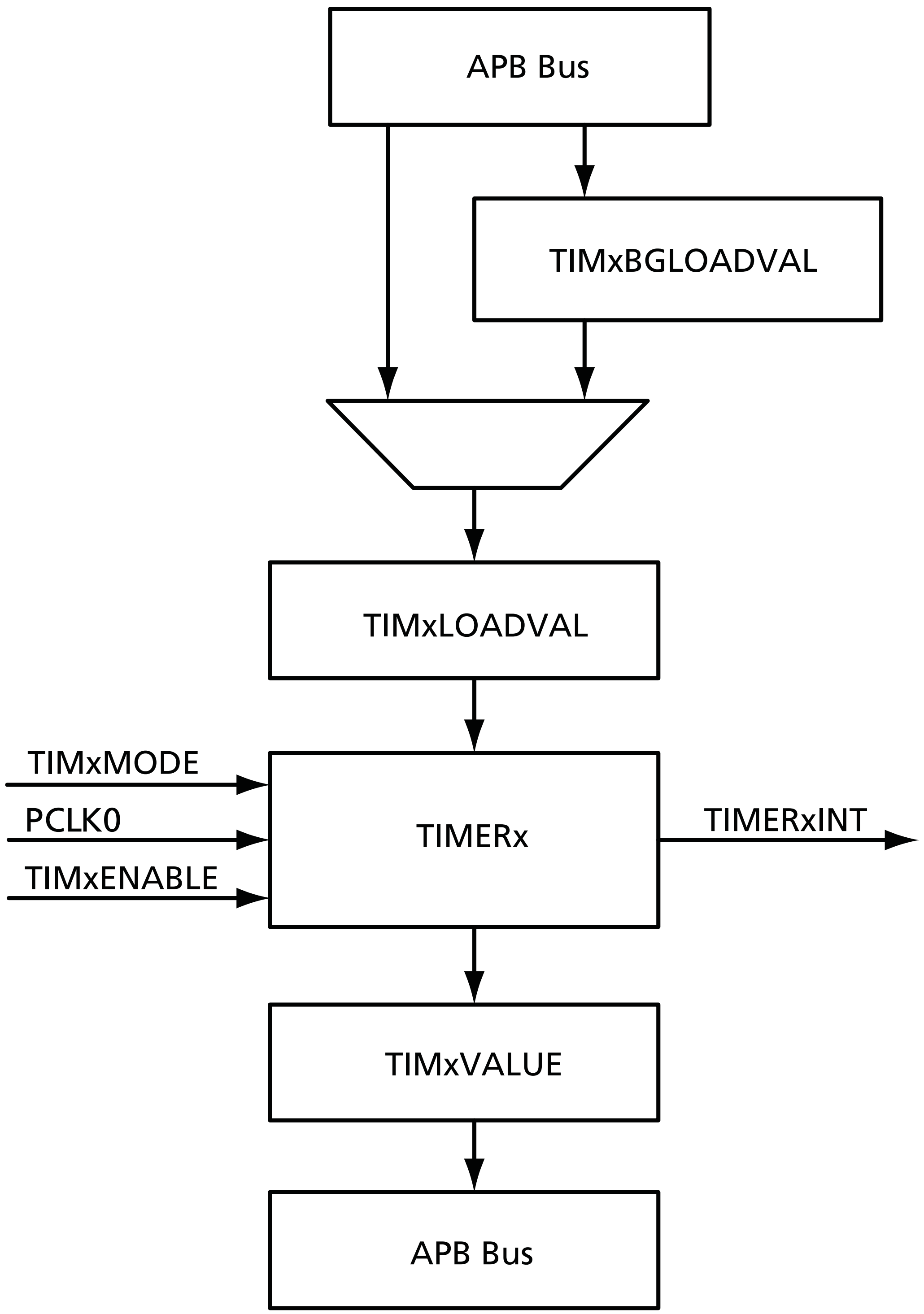
Timers on the SmartFusion (3)

- Real-Time Counter (RTC) System
  - Clocked from 32 kHz low-power crystal
  - Automatic switching to battery power if necessary
  - Can put rest of the SmartFusion to standby or sleep to reduce power
  - 40-bit match register clocked by 32.768 kHz divided by 128 (256 Hz)



Timers on the SmartFusion (4)

- System Timer
  - Two 32-bit timers that can be concatenated to one 64-bit timer
  - Clocked by PCLK0
  - One-shot or periodic interrupts
  - Load value defines upper bound



Interaction with the Outside World?

- Capture
  - Safe the time when a specific event happened, and signal an interrupt
- Compare
  - Generate an interrupt when counter reaches a specific value
  - Can set/reset/toggle a GPIO when counter reaches a specific value
- Pulse Width Modulated signal (PWM)
  - Special case of Compare
  - Set I/O when reaching a specific counter value
  - Clear I/O when reaching LOAD value
  - Usually used in continuous mode

- The SmartFusion is NOT a typical embedded MCU
- None of the timers has capabilities to interface with the outside world
- BUT: we have the FPGA fabric

Detailed View of Timer A on TI MSP430

- 16-bit Counter
  - Clock source selector
  - Dividers
  - Counter Register
  - Count Mode
    (up, down, up/down)
- Capture/Compare Unit
  - Capture Register
  - Compare Register
  - Capture/Compare Inputs
  - Interrupt
  - Output Unit

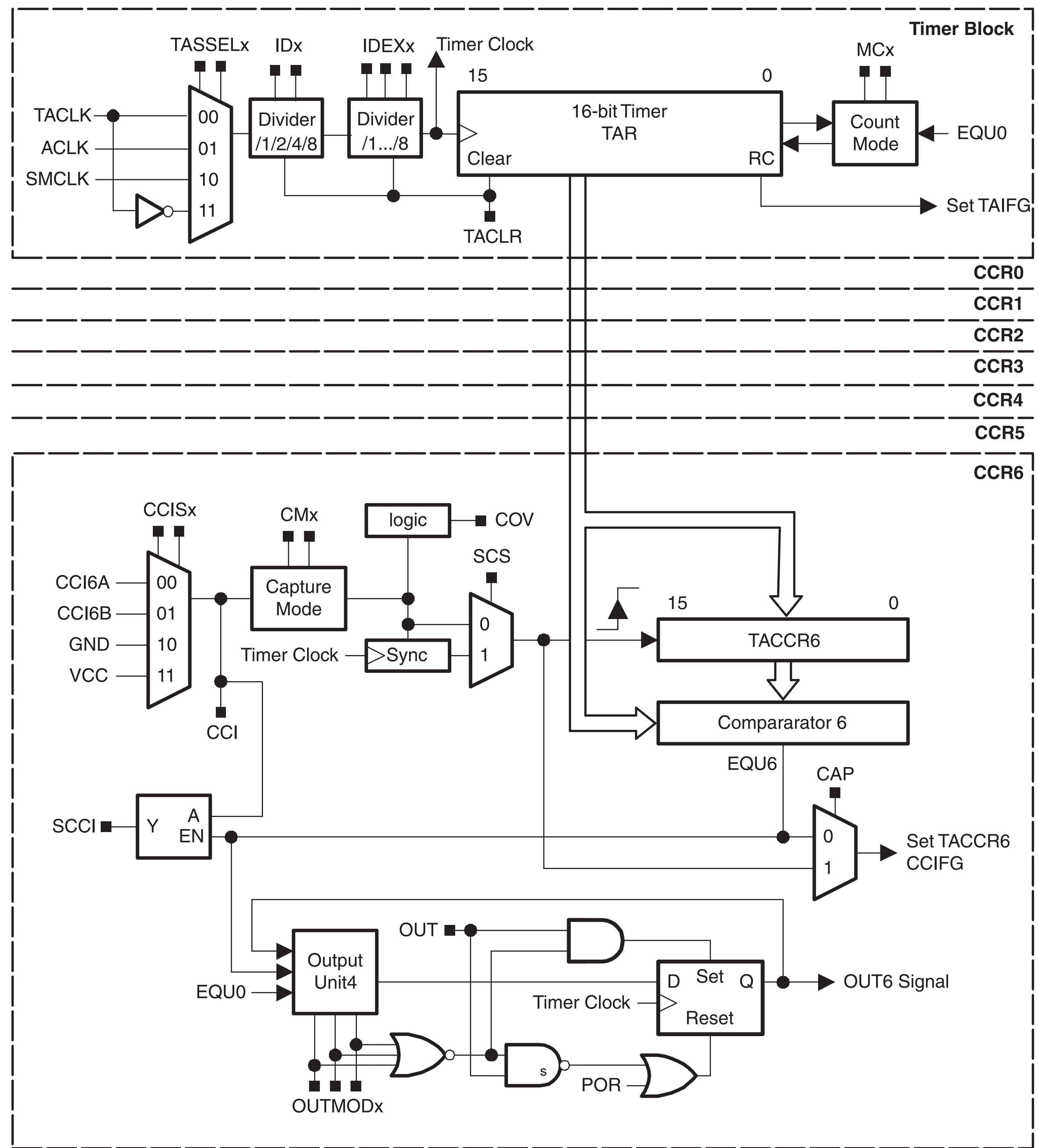
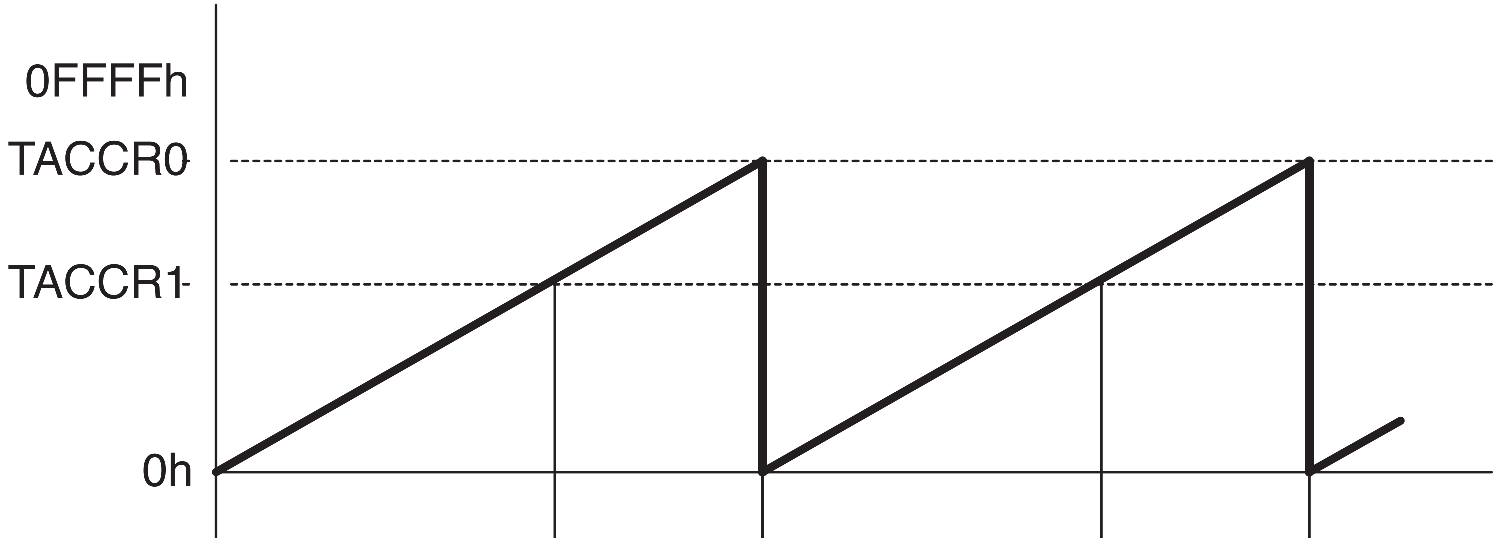


Figure 12-1. Timer_A Block Diagram

Timed Signal Generation (Timer UP mode)



Load

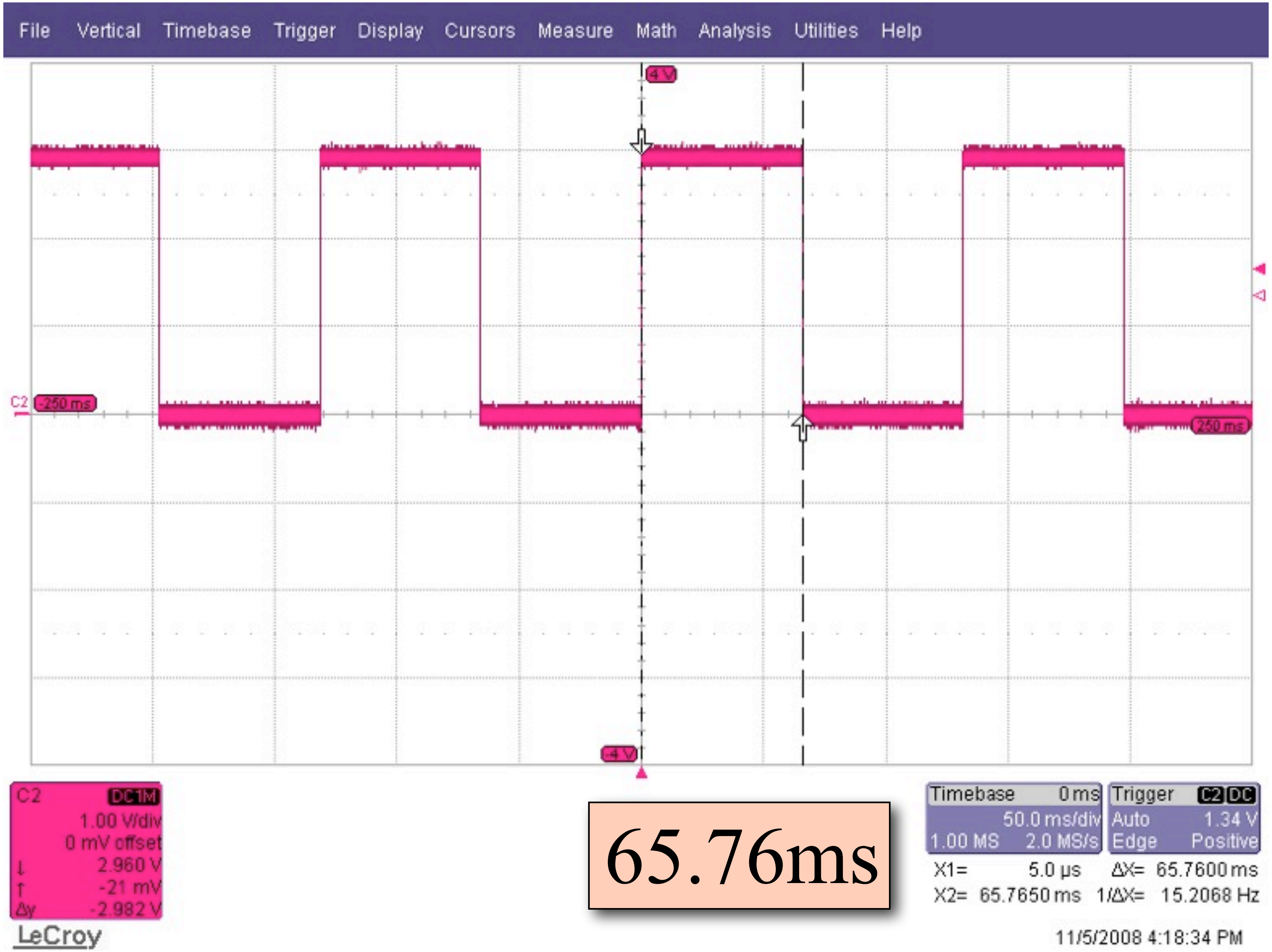
Compare

Example Code

- DCO at ~1.045MHz (on-chip RC oscillator of the MSP430)
- DCO clocks SMCLK

```
#include "msp430x54x.h"

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;                   // Stop WDT
  P1DIR |= 0x01;                              // P1.0 output
  TA1CCTL0 = CCIE;                            // CCR0 interrupt enabled
  TA1CCR0 = 50000;
  TA1CTL = TASSEL_2 + MC_2 + TACLR;           // SMCLK, contmode, clear TAR

  __bis_SR_register(LPM0_bits + GIE);         // Enter LPM0, enable interrupts
  __no_operation();                           // For debugger
}

// Timer A0 interrupt service routine
#pragma vector=TIMER1_A0_VECTOR
__interrupt void TIMER1_A0_ISR(void)
{
  P1OUT ^= 0x01;                              // Toggle P1.0
  TA1CCR0 += 50000;                           // Add Offset to CCR0
}
```

Example Output



Timer Virtualization

- What if we don't have enough hardware timers?
- Virtual timer library interface

```
typedef void (*timer_handler_t)(void);

/* initialize the virtual timer */
void initTimer();

/* start a timer that fires at time t */
error_t startTimerOneShot(timer_handler_t handler, uint32_t t);

/* start a timer that fires every dt time interval*/
error_t startTimerContinuous(timer_handler_t handler, uint32_t dt);

/* stop timer with given handler */
error_t stopTimer(timer_handler_t handler);
```

Timer Virtualization (2)

```
typedef struct timer
{
    timer_handler_t handler;
    uint32_t        time;
    uint8_t         mode;
    timer_t*        next_timer;
} timer_t;

timer_t* current_timer;

void initTimer() {
    setupHardwareTimer();
    initLinkedList();
    current_timer = NULL;
}

error_t startTimerOneShot(timer_handler_t handler, uint32_t t) {
    // add handler to linked list and sort it by time
    // if this is first element, start hardware timer
}

error_t startTimerContinuous(timer_handler_t handler, uint32_t dt) {
    // add handler to linked list for (now+dt), set mode to continuous
    // if this is first element, start hardware timer
}

error_t stopTimer(timer_handler_t handler) {
    // find element for handler and remove it from list
}
```

Timer Virtualization (3)

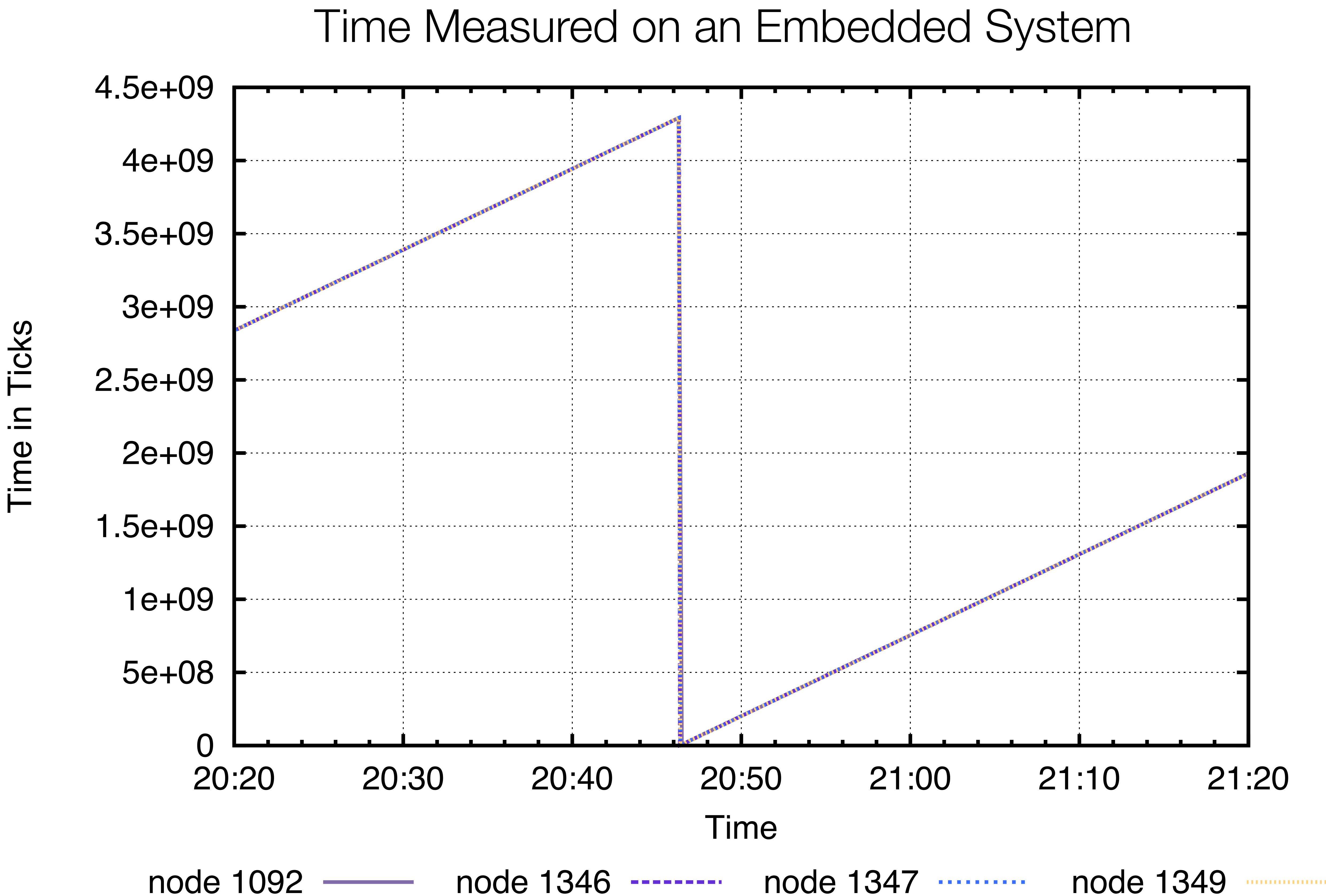
```
__attribute__((__interrupt__)) void Timer1_IRQHandler() {
    timer_t * timer;
    MSS_TIM1_clear_irq();
    NVIC_ClearPendingIRQ( Timer1_IRQn );
    timer = current_timer;

    if( current_timer->mode == CONTINUOUS ) {
        // add back into sorted linked list for (now+current_timer->time)
    }

    current_timer = current_timer->next_timer;

    if( current_timer != NULL ) {
        // set hardware timer to current_timer->time
        MSS_TIM1_enable_irq();
    } else {
        MSS_TIM1_disable_irq();
    }

    (*timer->handler))(); // call the timer handler

    if( timer->mode != CONTINUOUS ) {
        free(timer); // free the memory as timer is not needed anymore
    }
}
```

More Generic Real-Time Counters (RTC)

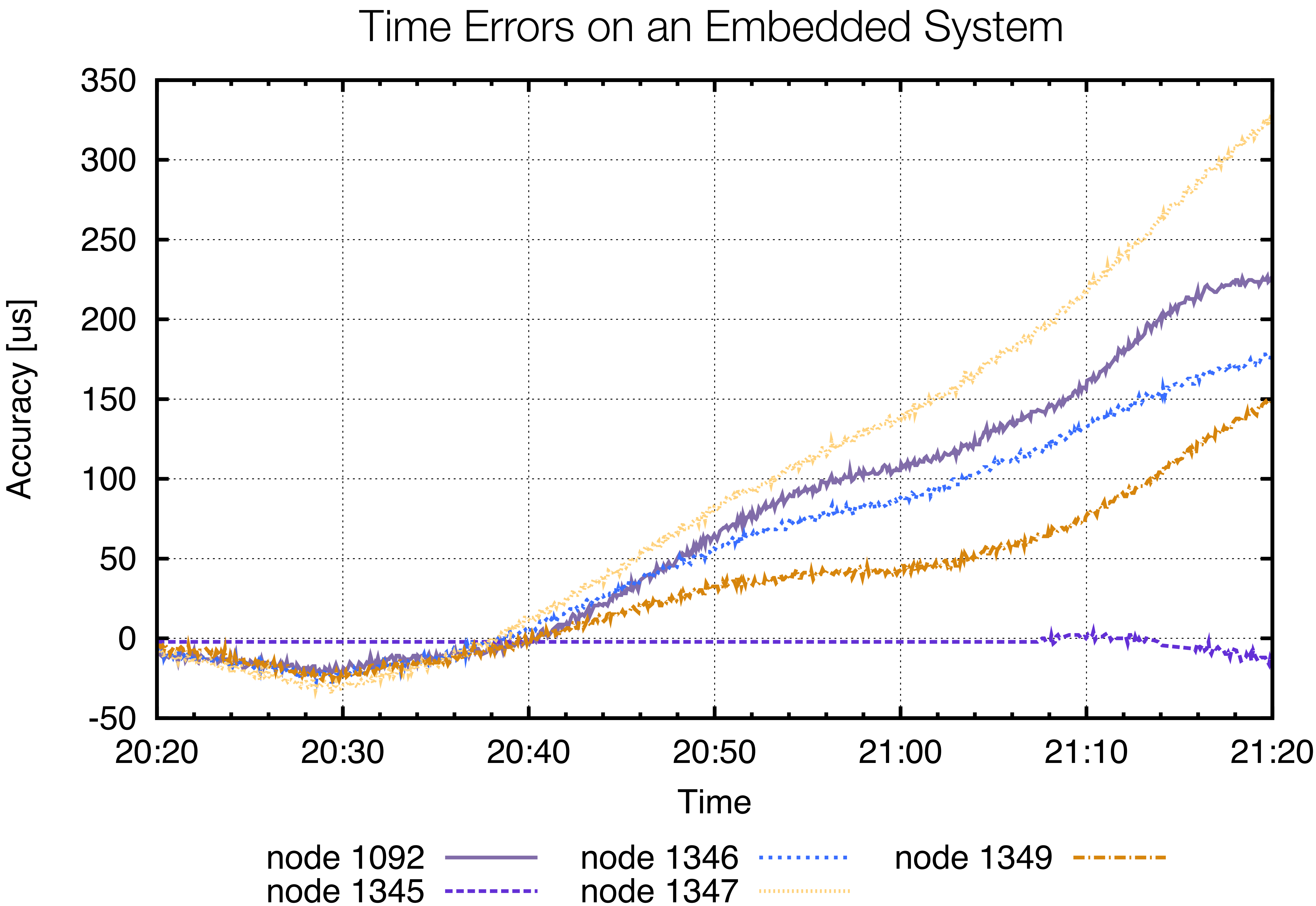
- Often provide a calendar function
- Example:
  - Maxim DS3231: Extremely Accurate I2C-Integrated RTC/TCXO/Crystal
- Accuracy
  - ±2ppm from 0°C to +40°C
  - ±3.5ppm from -40°C to +85°C
- Battery Backup Input for Continuous Timekeeping
- Low-Power Consumption (< 3.5 uA while outputing 32 kHz clock)
- Real-Time Clock
  - Counts Seconds, Minutes, Hours, Day, Date, Month, and Year
  - Leap Year Compensation Valid Up to 2100
- Two Time-of-Day Alarms
- Programmable Square-Wave Output
- Fast (400kHz) I^2C Interface
- 3.3V Operation
- Digital Temp Sensor Output: ±3°C Accuracy
- Register for Aging Trim

Clock accuracy and stability

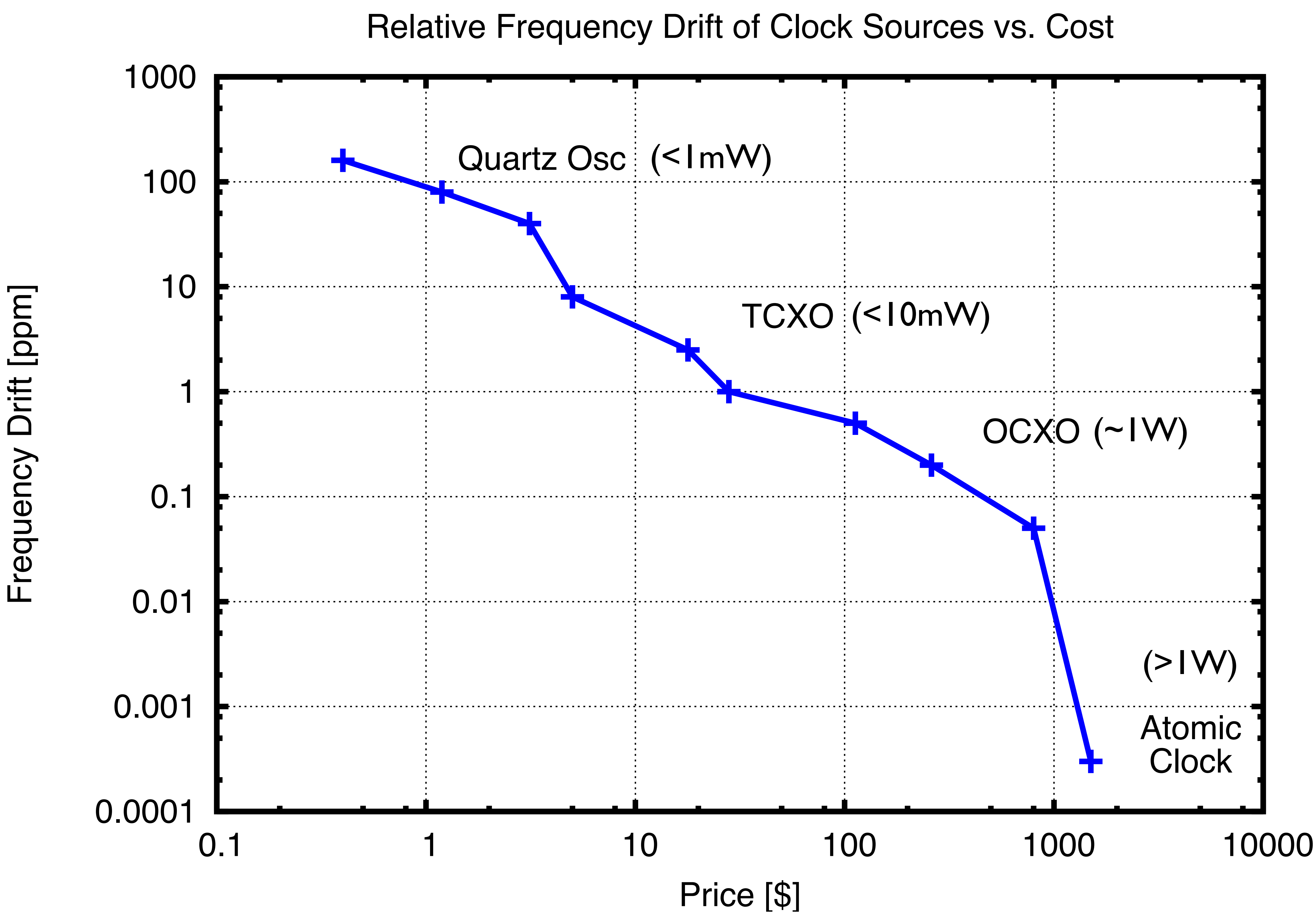
Example
4 clocks, 32 kHz clock, 32-bit counter



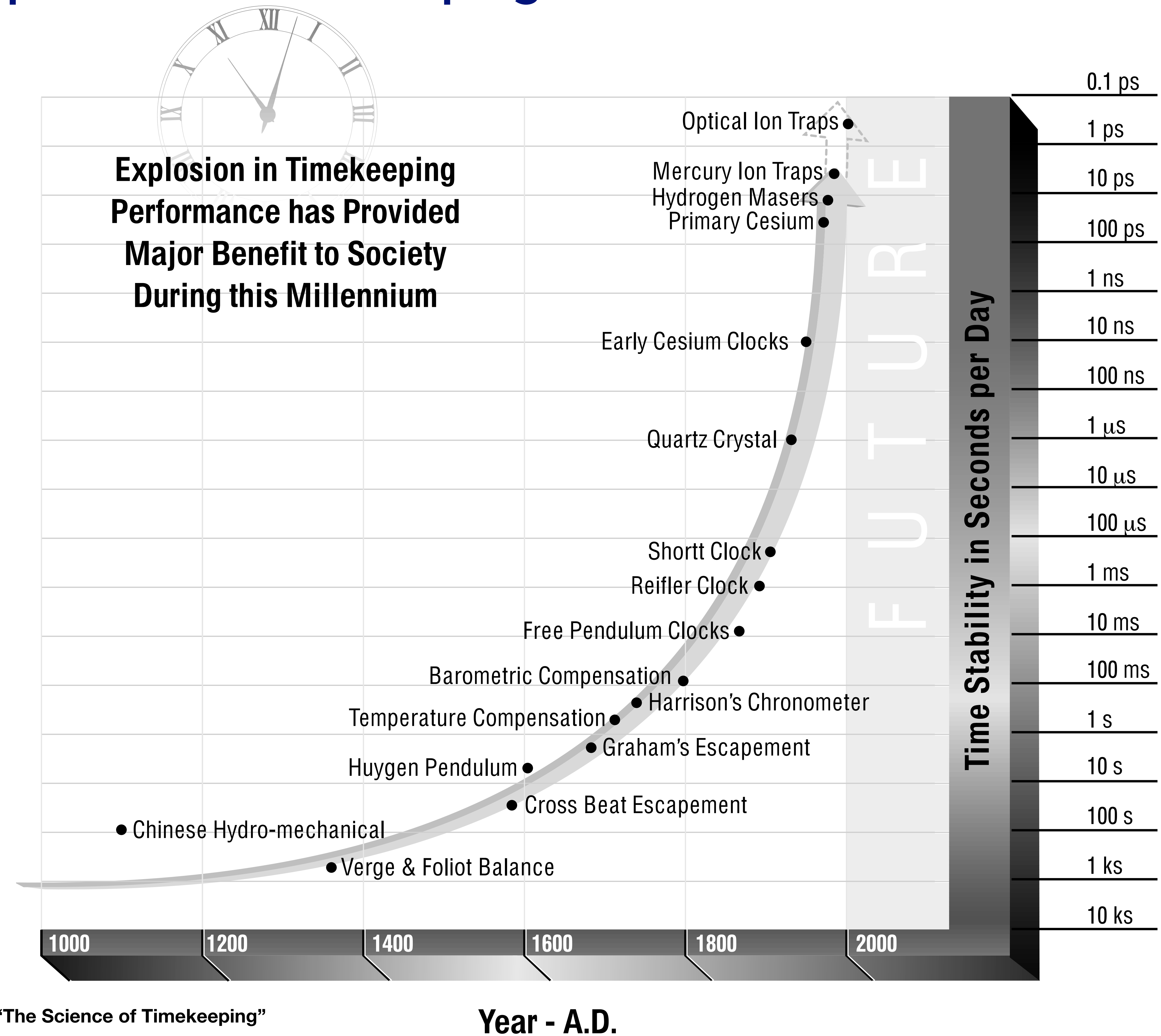
Example Errors



Resonating Elements



Explosion in Timekeeping

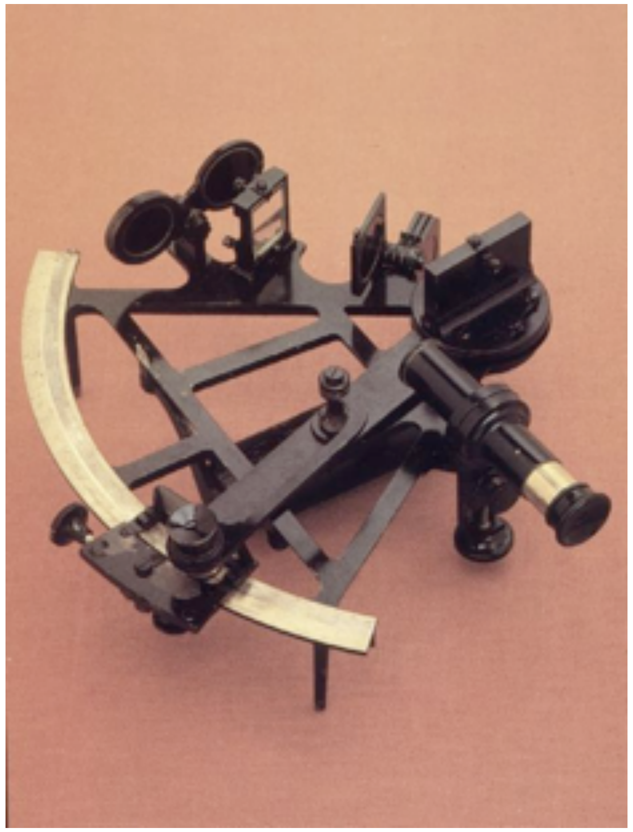


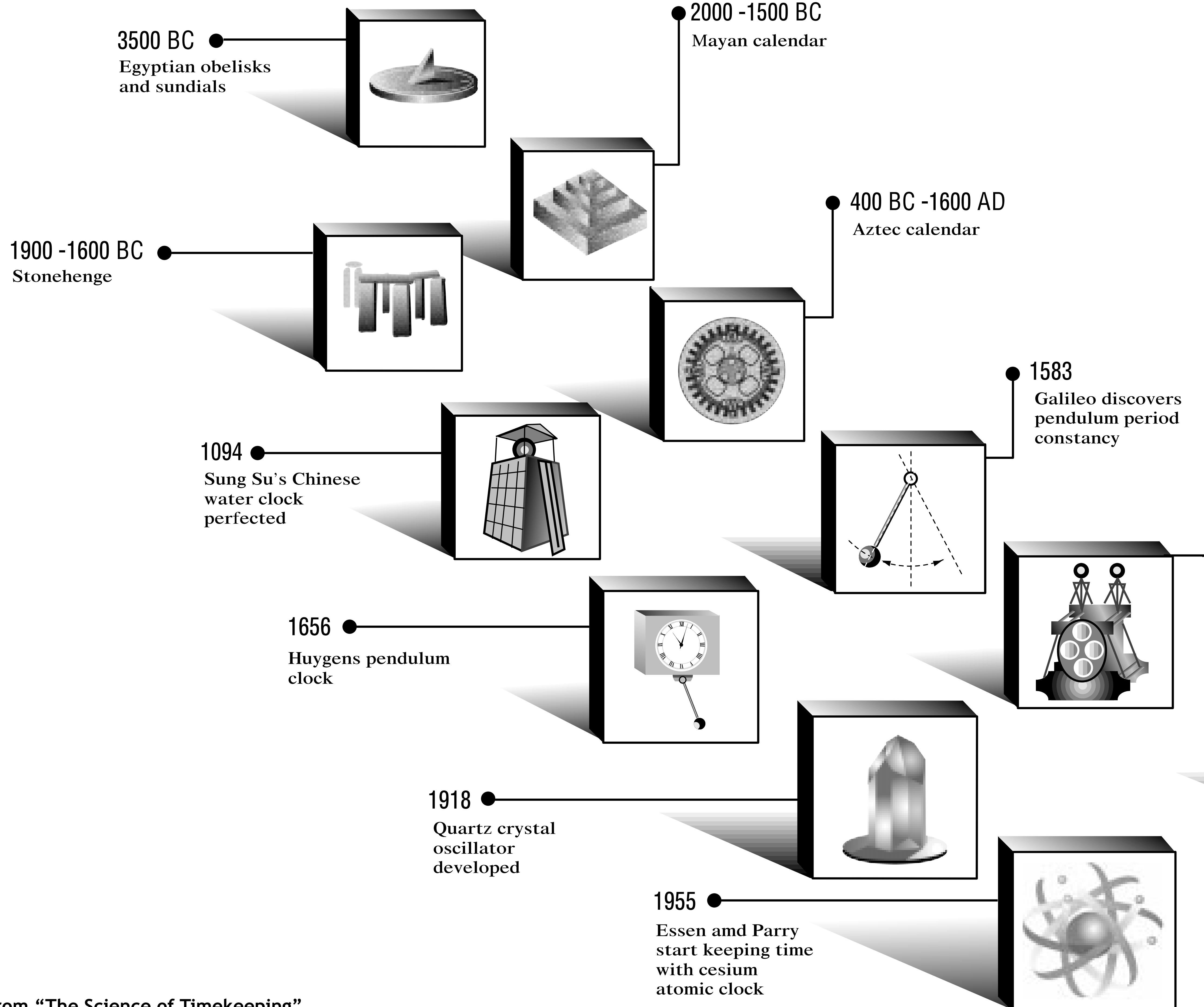
from "The Science of Timekeeping"

A short history of time

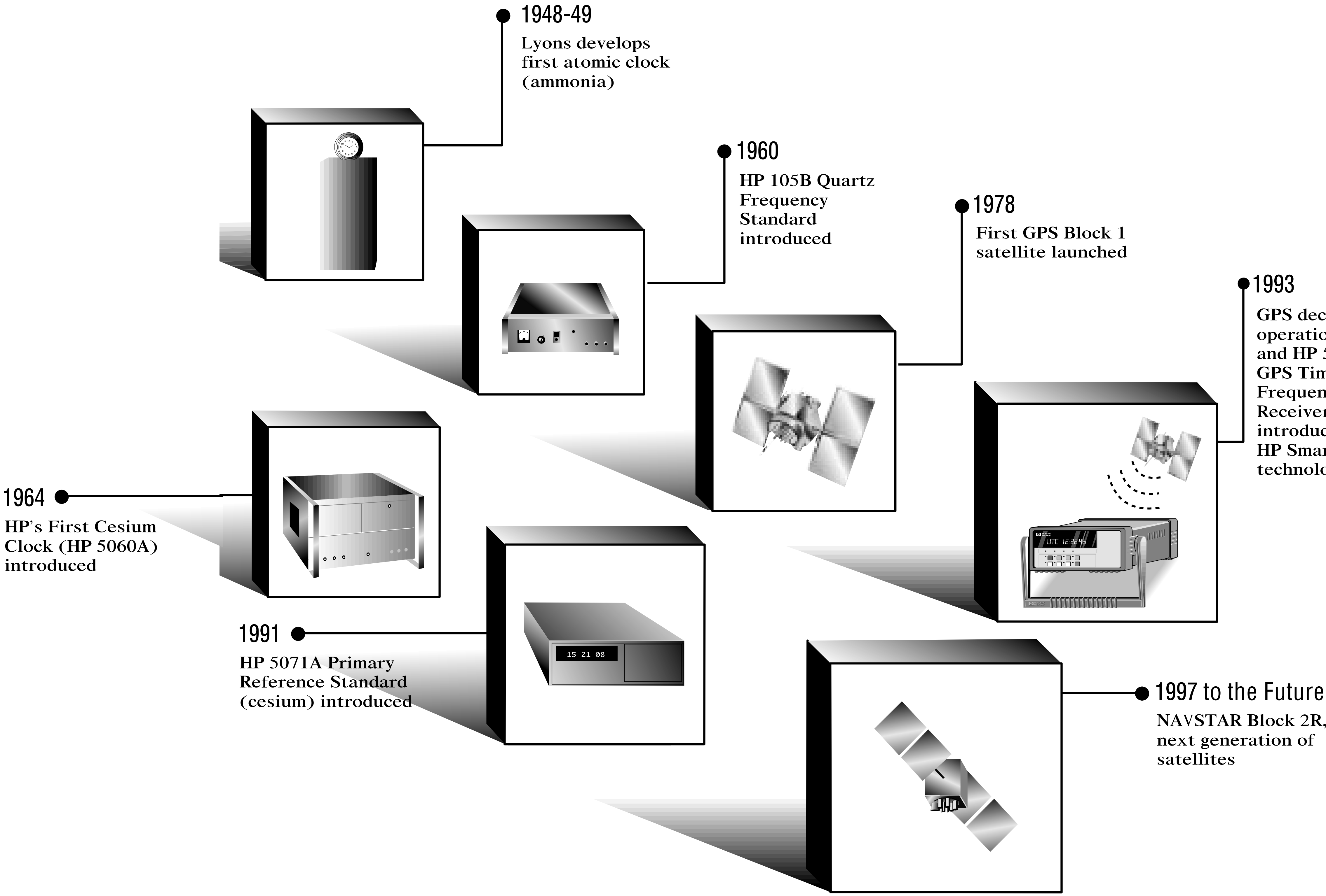
Time

- Why do we need to measure or know time?
  - Meeting times, lunch hours, office hours, opening hours
- In the 15th Century, naval exploration navigation drove time accuracy research
  - Latitude could be found with sextant by measuring the position of the sun at midday, or stars at night
  - Longitude is more difficult. You need sextant and accurate time
- 1714, British government established “The Board of Longitude”
  - £20’000 ($2,000,000 today) was offered to the person who could localize a ship within 30 nautical miles
  - This needed a clock that could keep time to within 3 seconds per day.

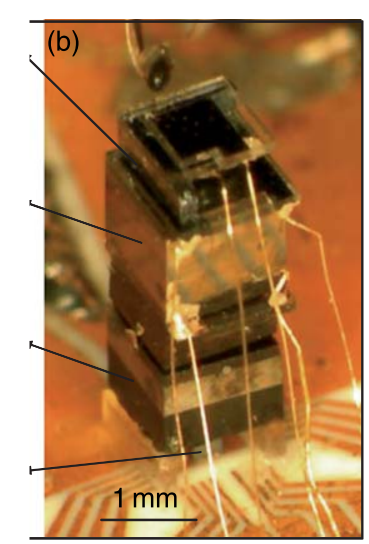




from “The Science of Timekeeping”



2004 Chip Scale Atomic Clocks



from "The Science of Timekeeping", "A Microfabricated Atomic Clock," 2004.

Time Fundamentals

- The most accurate measurement to humans is the second
- 1s = Time a cesium atom needs for 9,192,631,770 state transitions at 0°K
- Most accurate clocks can keep time to ±0.3ns, equivalent to ±1 second in 10 million years
- Many other measurements are defined from the second
  - "The length of the path travelled by light in vacuum during a time interval of 1/299,792,458 of a second (17th CGP, 1983, Resolution 1)"
- International Time Standard: UTC (Coordinated Universal Time)
  - UTC is based on the International Atomic Time (TAI) with leap seconds added
  - TAI is a weighted average of about 300 atomic clocks