Post project ideas on Ed
You can also come to office hours to brainstorm ideas

09: Clocks and Timers/Counters

Today

Where we've been:

I/O Peripherals, interrupts, embedded architecture

Where we're going:

Time - clocks, timers, watchdogs

Brief introduction to scheduling (execution time, concurrency)

Keeping track of time: system clocks

Or “oscillators”

Basis of control of a CPU - instructions happen on “edges” of a clock (**why?**)

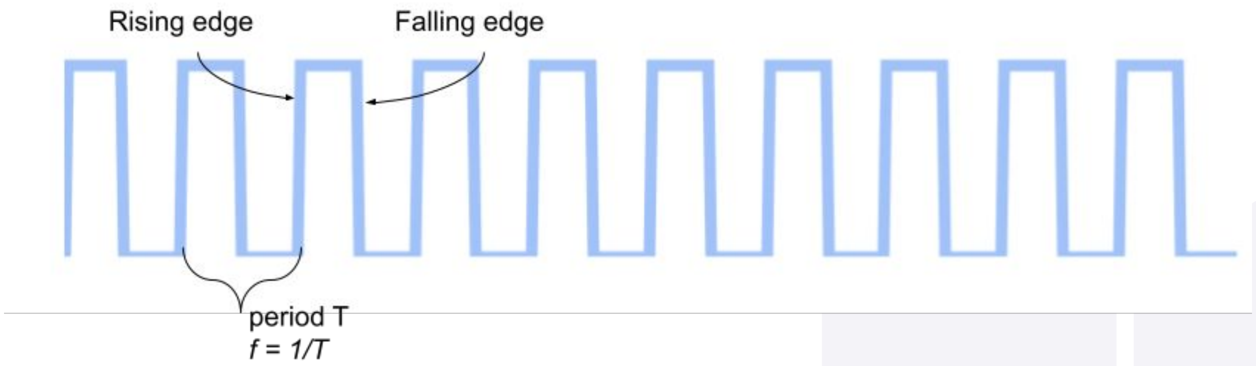
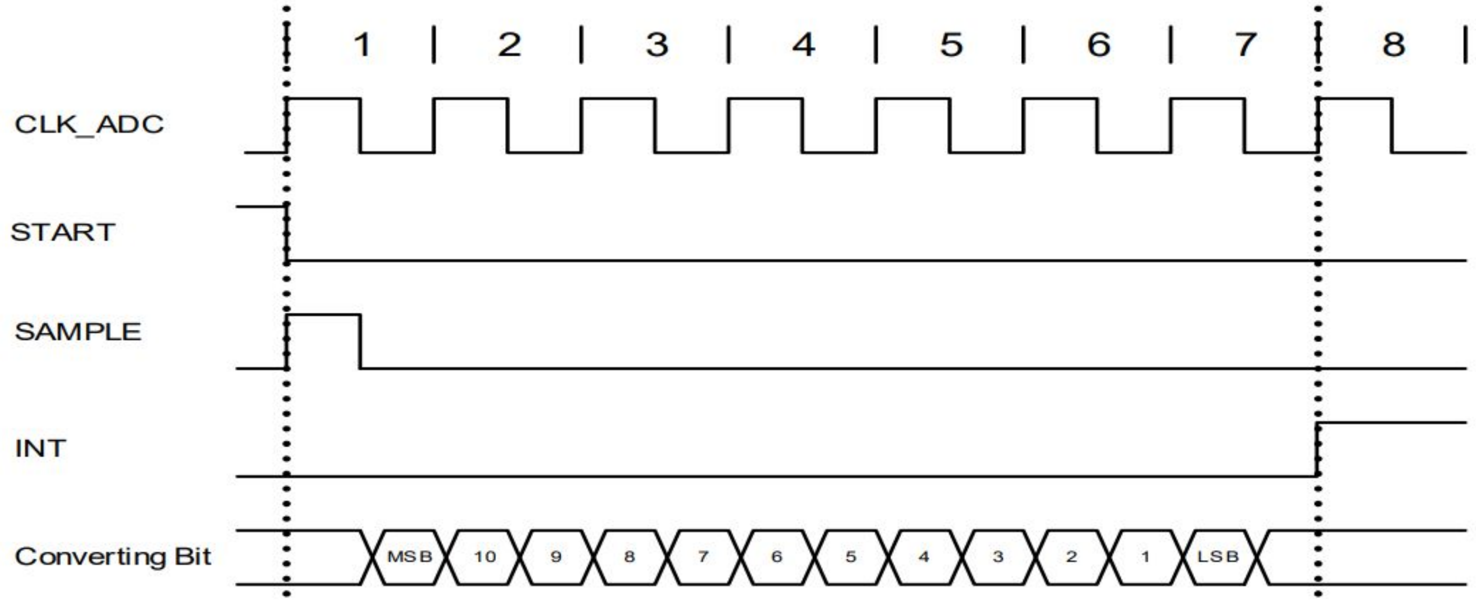


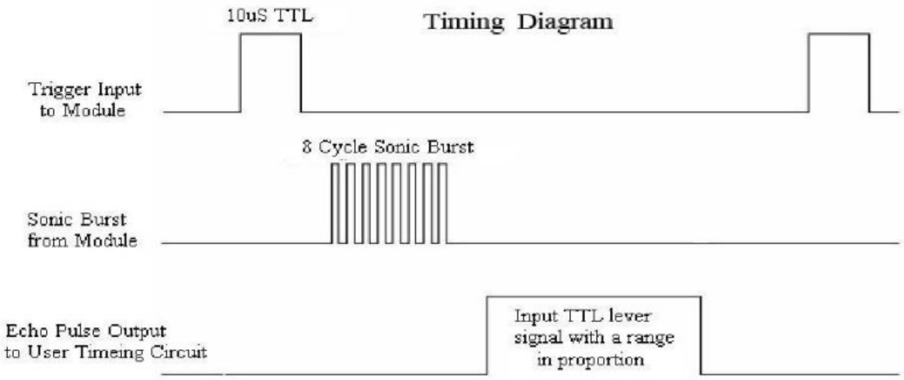
Figure 33-3. ADC Timing for One Conversion in Differential Mode without Gain



Timing Diagrams

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



Ultrasonic Proximity Sensor

Counting time

Most basic way to keep track of time on a CPU: # of clock ticks

On an 8MHz CPU: 8 million clock ticks = 1 second

What is the largest unit of time we can keep track of in 32 bits on an 8MHz clock?

$$\frac{2^{32} \text{ ticks}}{8 \times 10^{6} \frac{\text{ticks}}{s}}$$

How do we keep track of longer time periods?

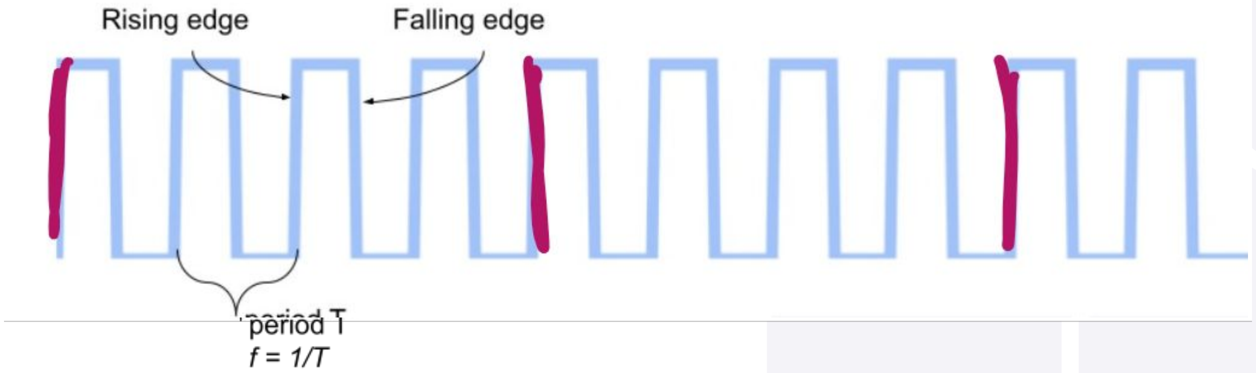
Timers

Keep track of time by incrementing every *n* clock ticks

On MCUs: hardware support

Often called something like TC (timer/counter) peripheral

Prescale the clock (divide it by 2, 4, 8…) and increment on the clock ticks



Uses for timers

- Count to a specific number of clock ticks and generate an interrupt (you will do this in lab!)
  - How Arduino keeps track of time for millis()
- Check for rollover and use this as a low-overhead way to measure time
  - Rollover: tick count reaches max value
  - Detected using polling or interrupt

Timer rollover math

48 MHz clock

Count every rising edge

32 bits: when will rollover happen?

*every 2^32 / (4 * 10^6) s*

Keeping track of time without using floating point

Keep track of fractional seconds (say every 2^{-16} seconds)

- Precompute how many fractional seconds between each rollover
- Increment by that many fractional seconds in a variable

Quantization margins

With perfect timekeeping, # of fractional seconds expected in a day:

5,662,310,400

48 MHz clock, pre-scaled by 16, 8 bit counter

- Effective frequency: 3 MHz
- Rollover every ~0.0000853 (2^8 / (3 * 10^6) seconds
- = every ~5.59 fractional seconds (~= 6)
- Rollovers in a day: 1,012,500,000
- Fractional seconds counted: 6,075,000,000
- Error: 7.3%

Clock drift

Imagine 32.768 kHz clock (common oscillator frequency - the SAM D21 has them too!)

0.001% drift rate (0.00001 seconds/second)

Drift during a day: 0.864 s

Drift during a year: 315.56 s

When would you want to use a slower clock? A faster clock?

An 8-bit, 16-bit, or 32-bit counter?

Summary

MCU architecture provides:

- Clocks of different frequencies
- Pre-scaler constants for timers/counters
- Registers to count clock ticks
- Ability to detect timer/counter events such as rollover

calculate elapsed time based on these configurations