TI-RSLK MAX

Texas Instruments Robotics System Learning Kit

Texas Instruments

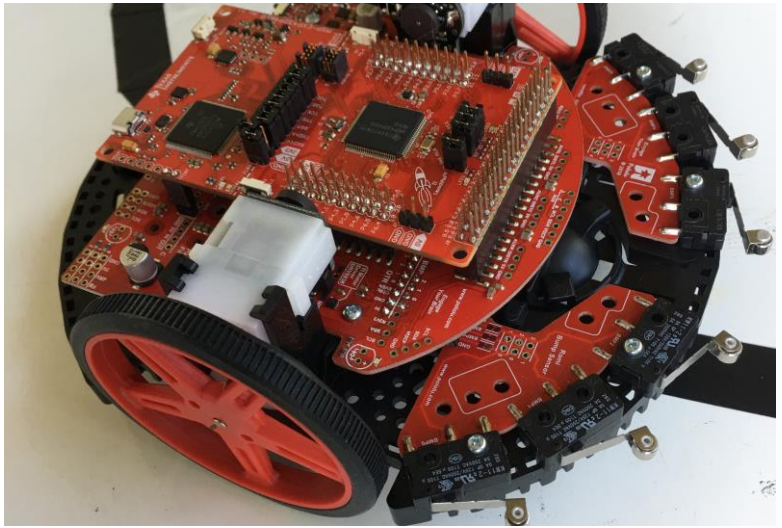
Module 9

Lecture: SysTick Timer - Theory

SysTick Timer

You will learn in this module

- Fundamentals of SysTick Timer
- Measure elapsed time
  - Precision
  - Range
  - Resolution
- Software delay



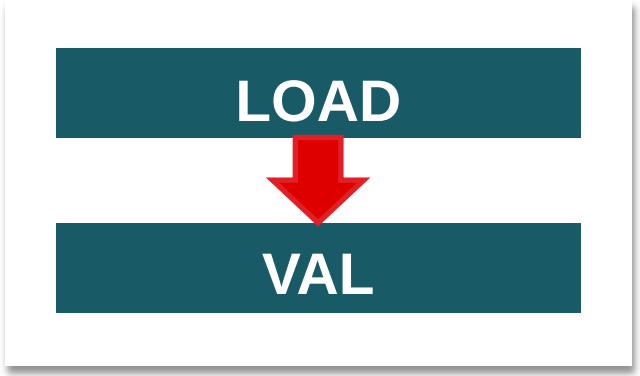
SysTick Timer :

SysTick performs Timer/Counter operation in all ARM

- Create time delays
- Generate periodic interrupts

How it works

- 24-bit down counter decrements at bus clock frequency
- With a 48 MHz bus clock, decrements every 20.83 ns
- Software sets a 24-bit LOAD value of n
- The counter, VAL, goes from n → 0
  - Sequence: n, n-1, n-2, n-3… 2, 1, 0, n, n-1…
- SysTick is a modulo n+1 counter:
- VAL = (VAL - 1) mod (n+1)



SysTick Timer Initialization

Flag

Bus clock

Turn on

31-24	23-17	16	15-3	2	1	0	Name
0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	SysTick->CTRL
0	24-bit RELOAD value						SysTick->LOAD
0	24-bit CURRENT value of SysTick counter						SysTick->VAL

Table 9.0 SysTick Registers

```
void SysTick_Init(void){
  SysTick->LOAD = 0x00FFFFFF;
  SysTick->CTRL = 0x00000005;
}
```

At 48 MHz, it rolls over about every 349ms

Measure Elapsed Time

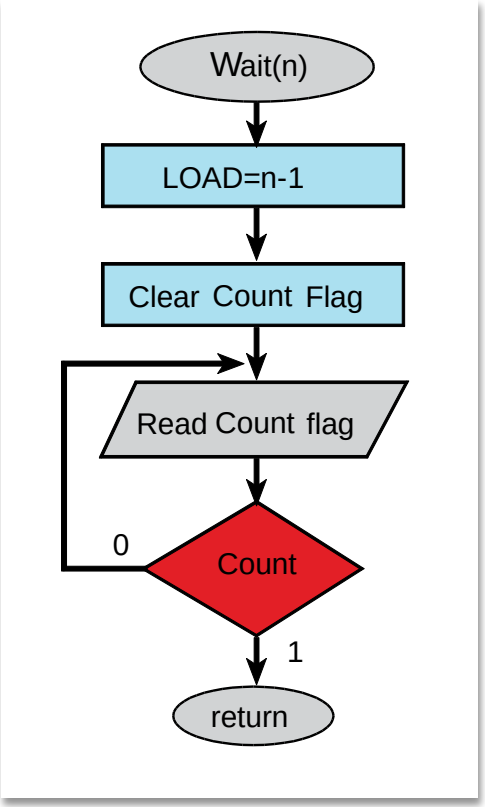
```
Start = SysTick->VAL;
SystemUnderTest();
Stop = SysTick->VAL;
Delta = 0x00FFFFFF&(Start-Stop);
```

Time to execute?

At 48 MHz

- 24-bit precision ← # of distinct measurements
- 20.83ns resolution ← Smallest change
- 349ms range ← Largest possible

SysTick Timer Wait

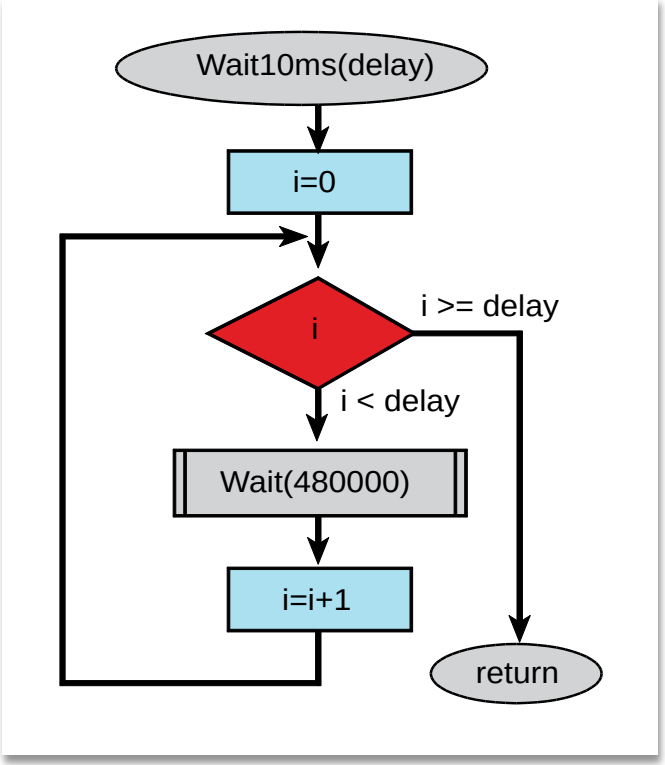



```
void SysTick_Wait(uint32_t n){
  SysTick->LOAD = n-1;
  SysTick->VAL = 0;     // clear Count
  while((SysTick->CTRL&0x00010000)== 0){};
}
```

Count is in bit 16

At 48 MHz, it works up to 349ms
Doesn't work for n=0 or n=1

SysTick Timer : Generate 10 ms Wait




```
void SysTick_Wait10ms(uint32_t delay){
  for(uint32_t i=0; i<delay; i++){
    SysTick_Wait(480000);
  }
}
```

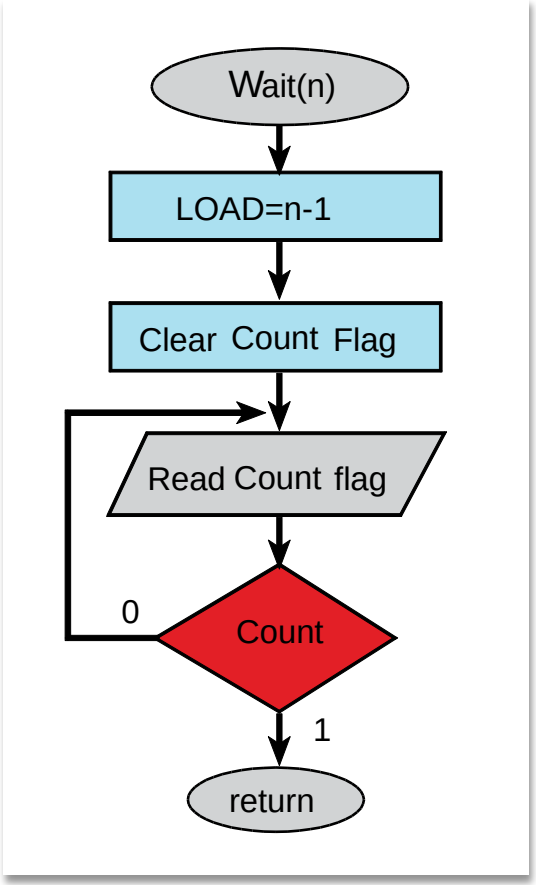
48 cycles is 1us

48,000 cycles is 1ms

480,000 cycles is 10ms

Summary

- SysTick is a built in timer
  - Measuring elapsed time
  - Creating software delay



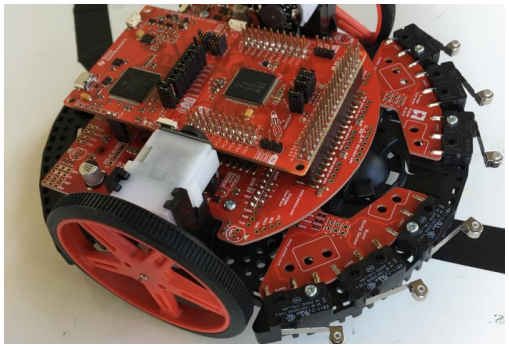
Module 9

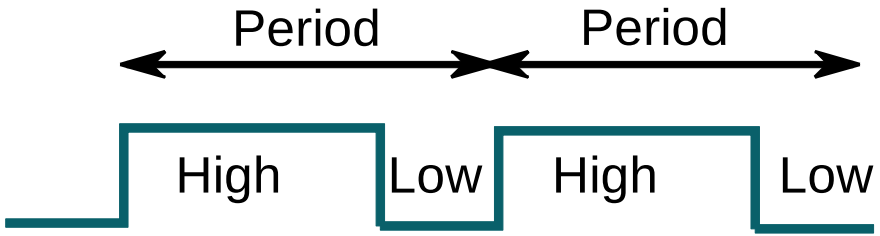
Lecture: SysTick Timer - PWM

SysTick Timer

You will learn in this module

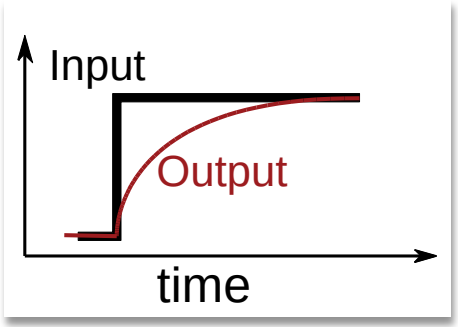
- Concept of **Pulse Width Modulation** (PWM) and Duty Cycle
- Create pulse width modulated (PWM) signals using SysTick Timer Delay
- Use PWM to control brightness of an LED
- Apply PWM to create digital to analog converter (DAC)





How fast is the device?

Change the input (step change)
Measure the output response



Out(t) = $A + Be^{-t/\tau}$

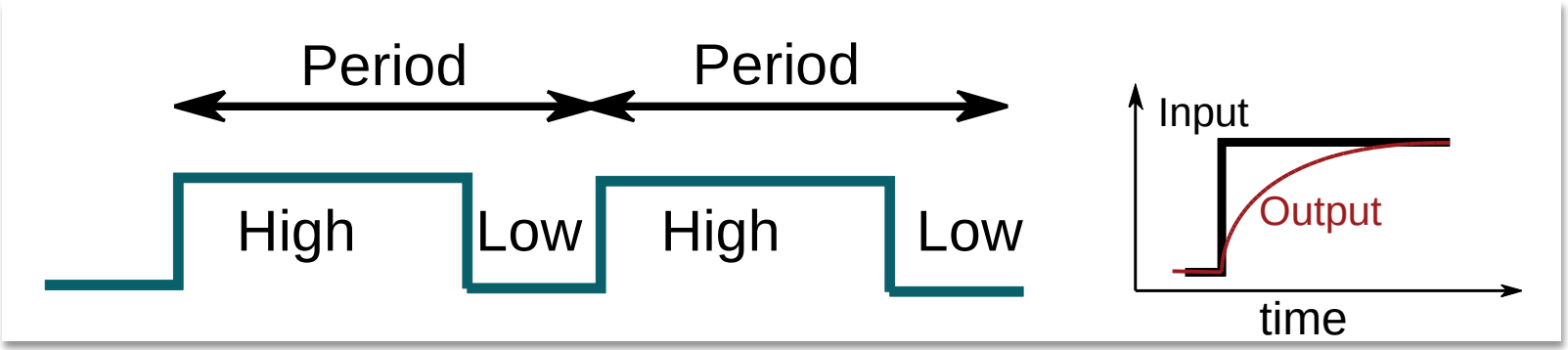
Time constant, τ, is the time to reach 0.63 of final

Time constant , τ	
HLMP-4700 LED	90ns
DC motor	100ms

Pulse Width Modulation

```
while(1){
  P1->OUT |= 0x01;    // red LED on
  SysTick_Wait(High);
  P1->OUT &= ~0x01;   // red LED off
  SysTick_Wait(Low);
}
```

$$\text{Dutycycle} = \frac{\texttt{High}}{\texttt{High}+\texttt{Low}} = \frac{\texttt{High}}{\texttt{Period}}$$

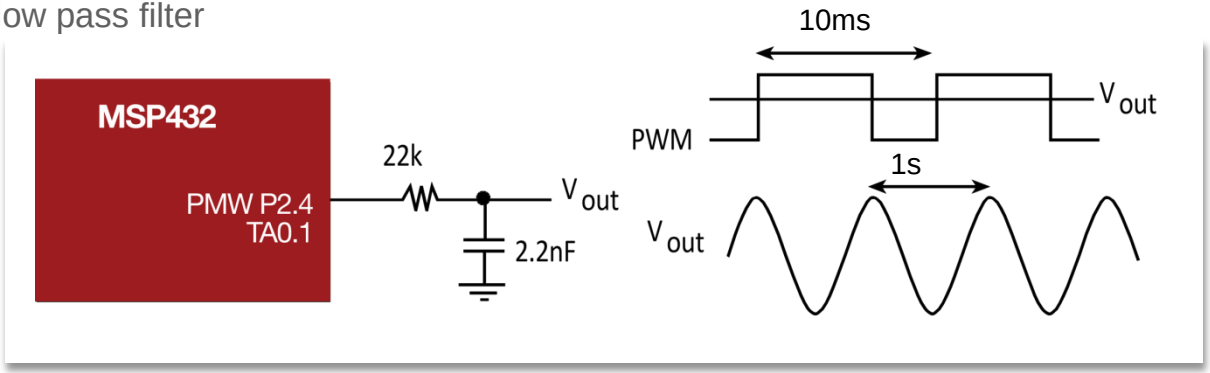


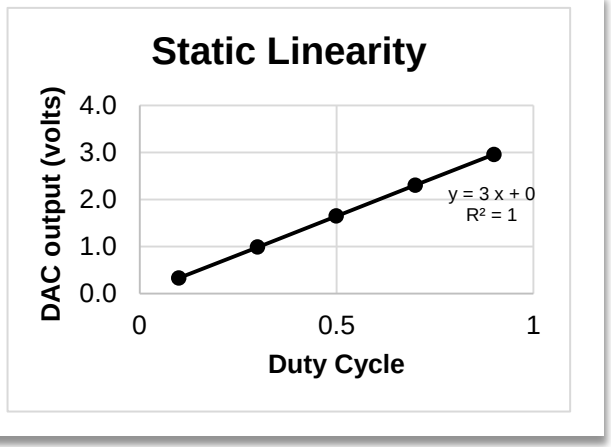
`High+Low` is a constant

If fast enough, the device responses to the average

PWM - Digital to Analog Converter

PWM + low pass filter





LPF cutoff:

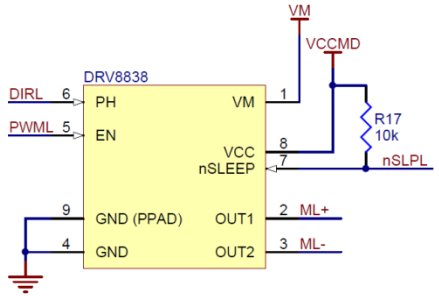
$f_c = 1/(2\pi RC)$

f_c > analog wave

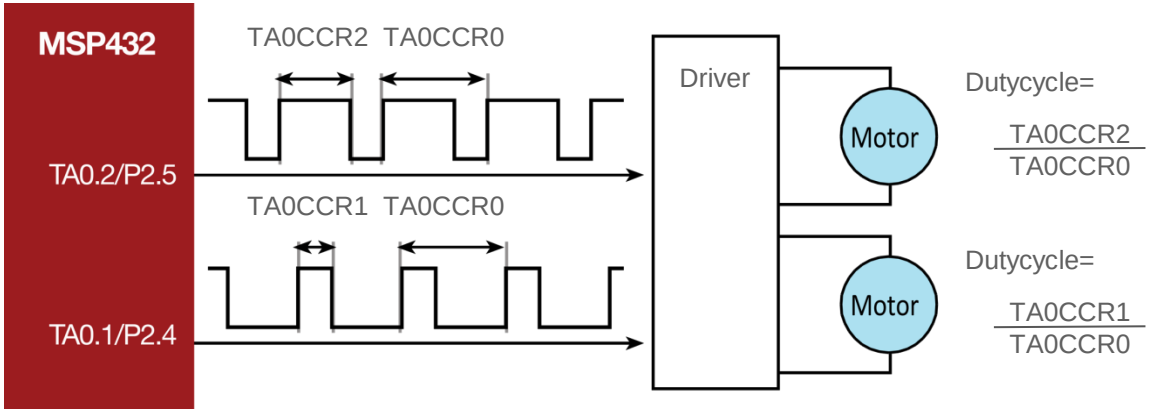
f_c < digital frequency

Applications of PWM:

- Control brightness of LEDS
- 120V /60Hz appliances
- Use it to make a DAC
- Transfer power to control motors



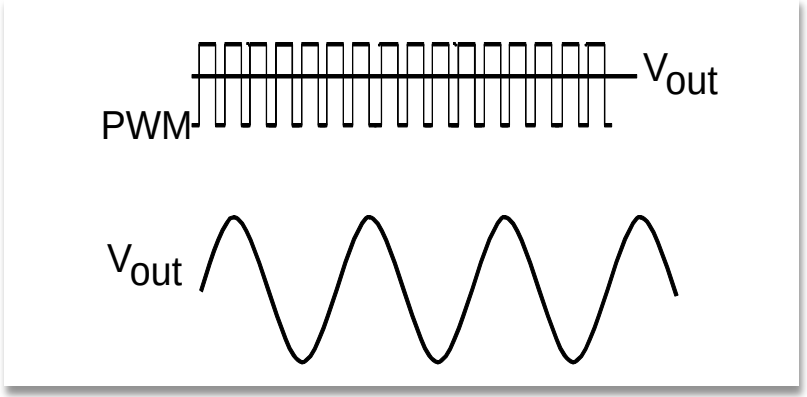
Motor driver TI DRV8838

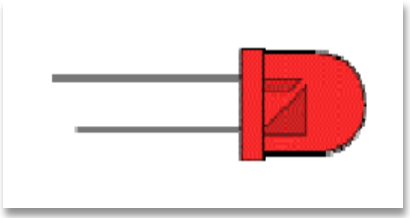


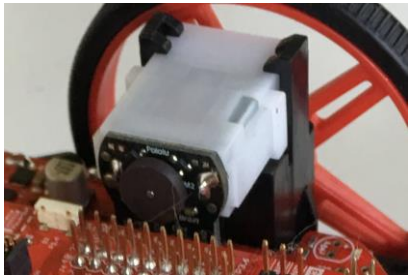
Interface TI Launchpad with Motor driver TI DRV8838 using PWM

Summary

- SysTick is a built in timer
  - Measuring elapsed time
  - Creating software delay
- PWM
  - Implemented with software delays (inefficient)
  - Choose the fixed frequency faster than the device
  - Device responds linearly to duty cycle
  - Provides for high precision outputs
- Applications
  - Dimming
  - DAC
  - Motors







ti.com/rslk

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265