

TI-RSLK

Texas Instruments Robotics System Learning Kit



TEXAS INSTRUMENTS



Module 8

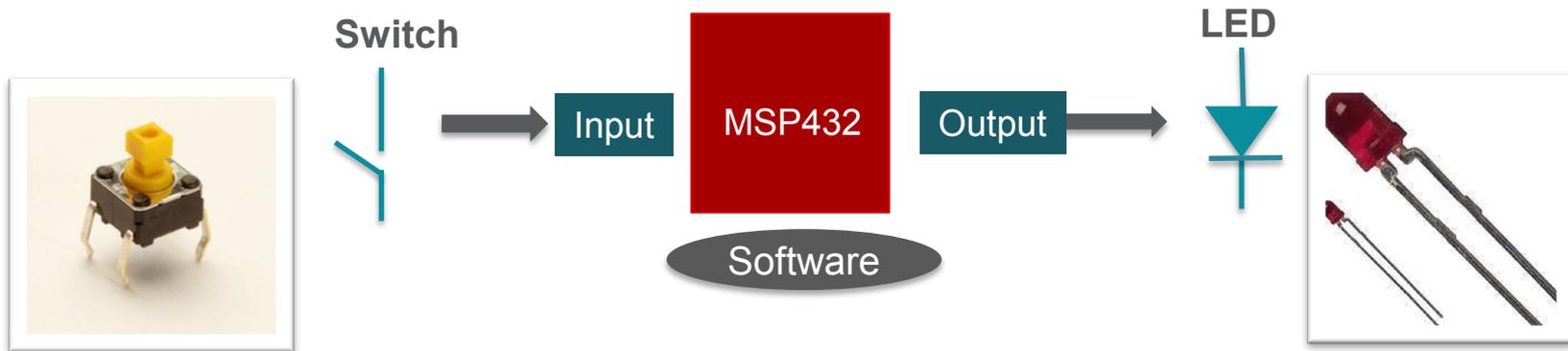
Lecture : Interfacing input and output - Switches



Interfacing input devices using Switches

You will learn in this module

- Fundamentals of switches
- How to interface switches TI's Launchpad Development board
- Software driver (set of functions to create an abstract module)
- Motivation for lab

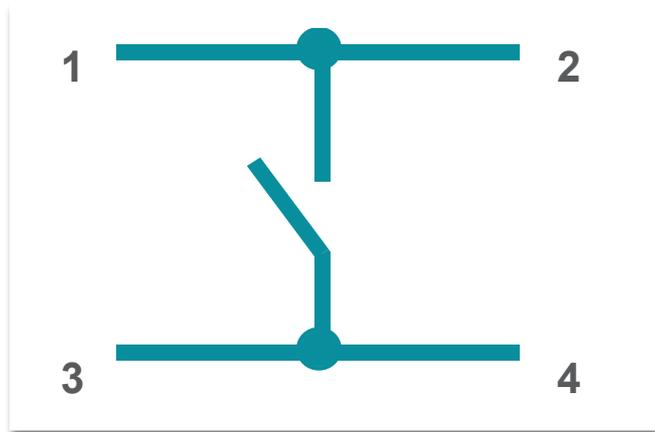




Switch Configuration

Not pressed $R = 100\text{M}\Omega$

Pressed $R = 0.1\Omega$



Not pressed

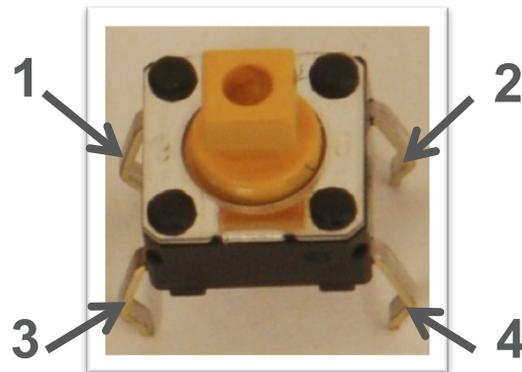
Pressed



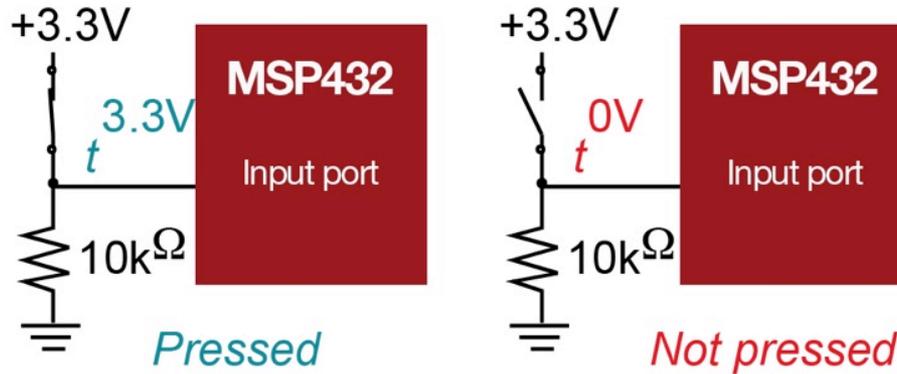
Open



Closed



Positive Logic Switch Interface

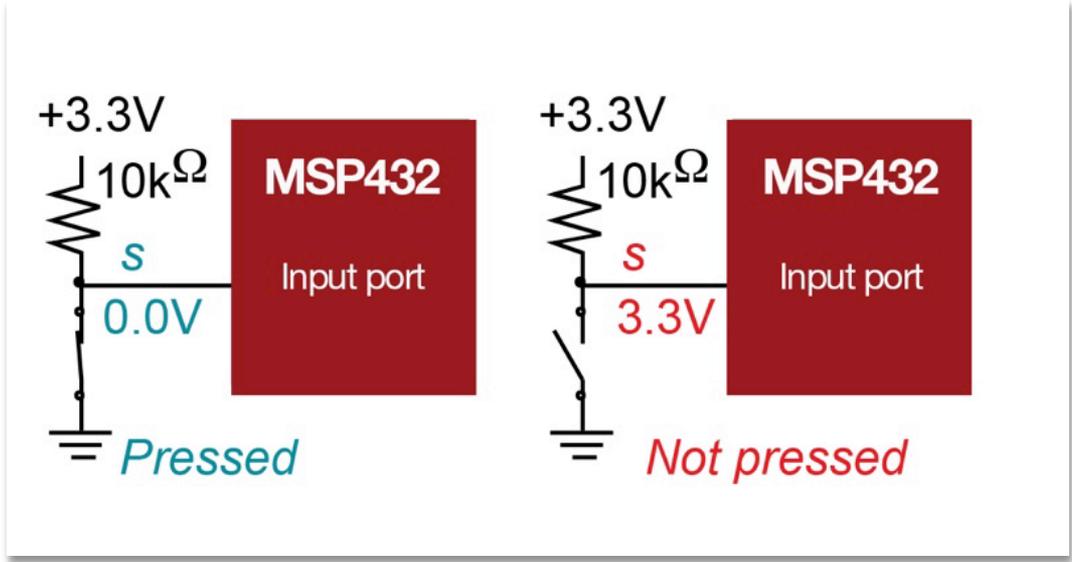


Positive Logic t

- pressed, 3.3V, true
- not pressed, 0V, false



Negative Logic Switch Interface

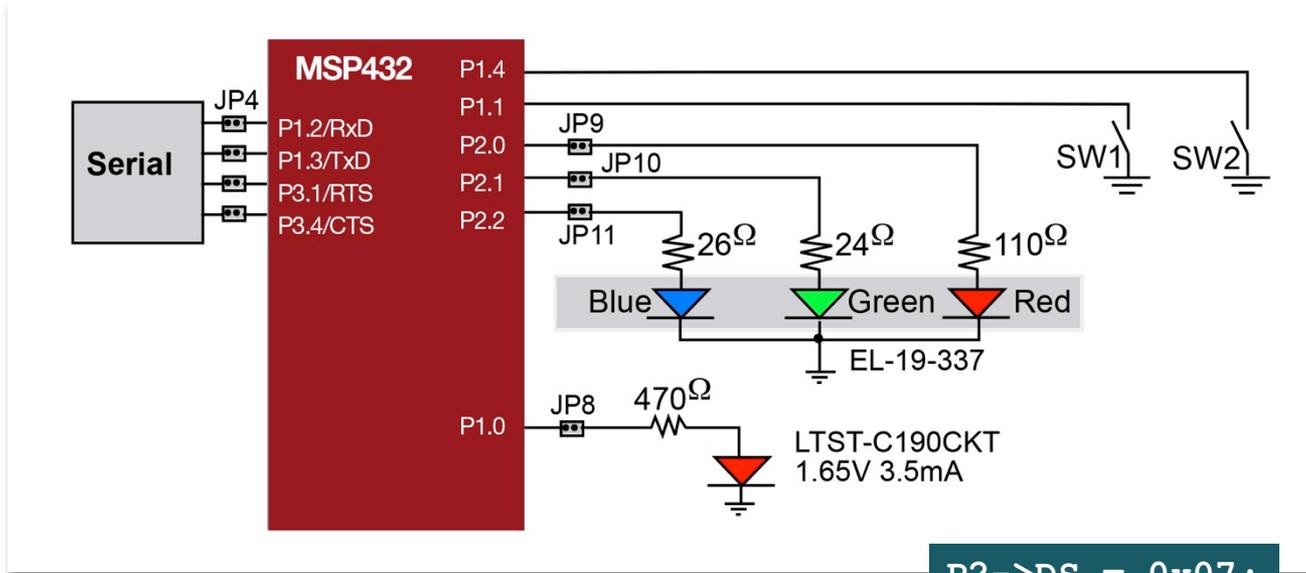


Negative Logic s

- pressed, 0V, true
- not pressed, 3.3V, false



LaunchPad Switches and LEDs



```
P2->DS = 0x07;
```

The Switches on the LaunchPad

- Negative logic
- Require internal pull-up

The LEDs are positive logic



Software Driver (inputs)

Initialization (executed once at beginning)

1. Set *DIR* to 0 for input
2. Enable pullup on inputs

Input from switches

1. Read from data input port
2. Mask (select) desired bits

```
all = P1->IN;  
in = all&0x01;
```

Mask



Software Driver (simple, not friendly)

```
#include "msp.h"

void Port1_Init(void){
    P1->DIR = 0x00;    // 1) make P1.4 and P1.1 in
    P1->REN = 0x12;    // 2) enable pull resistors on P1.4 P1.1
    P1->OUT = 0x12;    // P1.4 and P1.1 are pull-up
}

uint8_t Port1_Input(void){
    return (P1->IN&0x12); // read P1.4,P1.1 inputs
}
```

See [InputOutput_MSP432](#) example project



Software Driver (friendly)

```
#include "msp.h"
void Port1_Init(void){
    P1->DIR &= ~0x12; // 1) make P1.4 and P1.1 in
    P1->REN |= 0x12; // 2) enable pull resistors on P1.4 P1.1
    P1->OUT |= 0x12; // P1.4 and P1.1 are pull-up
}
uint8_t Port1_Input(void){
    return (P1->IN&0x12); // read P1.4,P1.1 inputs
}
```

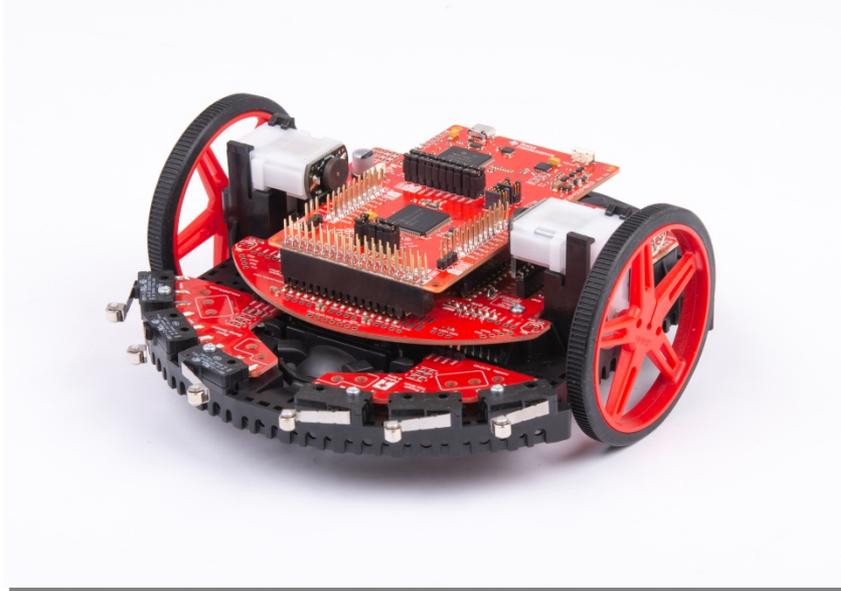
See [InputOutput_MSP432](#) example project



Application

Switches provide

1. Feedback to robot as bump sensors to determine if there is an obstruction
2. Control/command inputs to robot (e.g., start/stop)

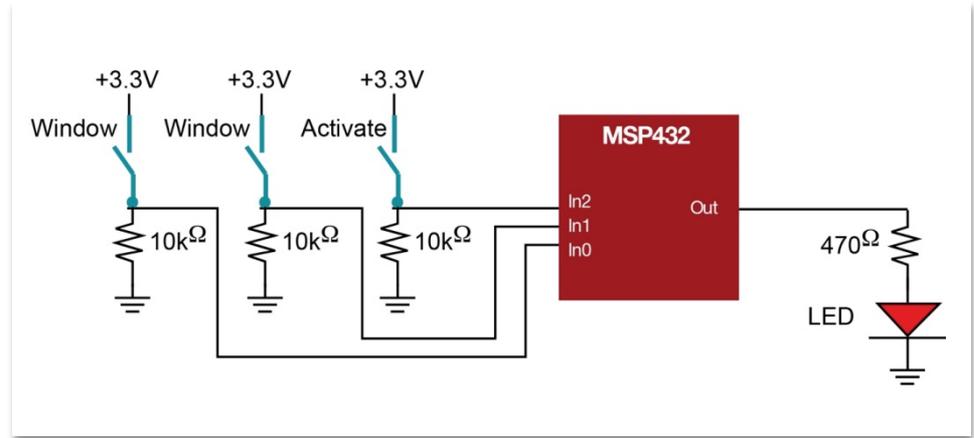




Summary

- Positive and negative logic
- Ohm's Law for resistors
- Switch interface with pullup or pulldown
- LaunchPad switches and LEDs
- Software driver
 - Initialization
 - Input/Output functions

$$V = I * R$$





Module 8

Lecture : Interfacing input and output - LEDs

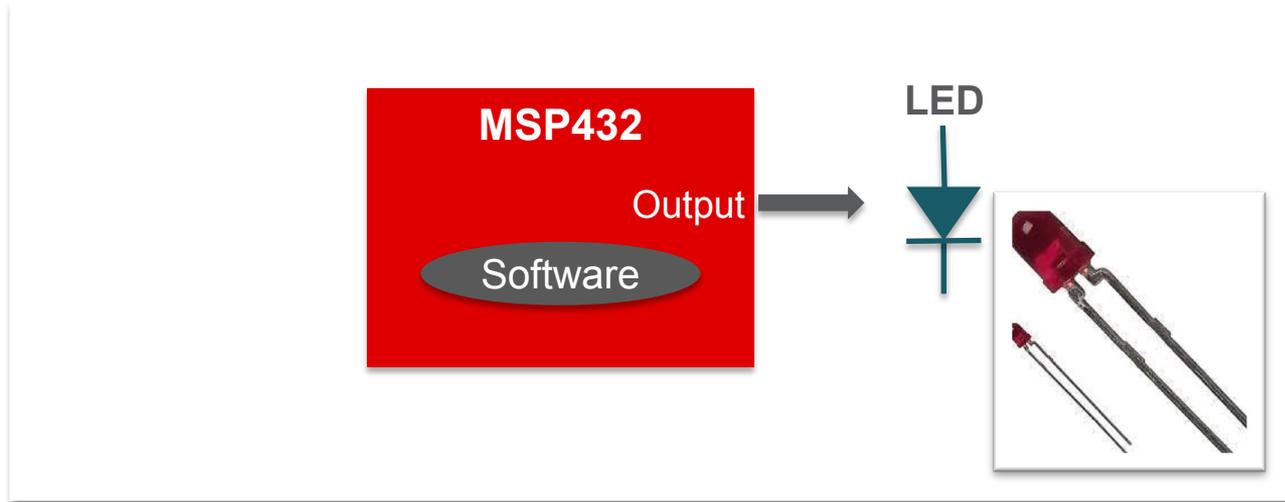


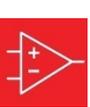
Lecture

Interfacing output devices using LEDs

You will learn in this module

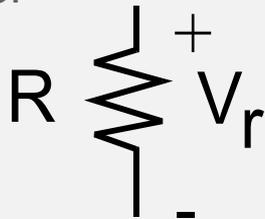
- Fundamentals of LEDs
- How to LEDs to TI's Launchpad Development board
- Software driver (set of functions to create an abstract module)
- Motivation for lab



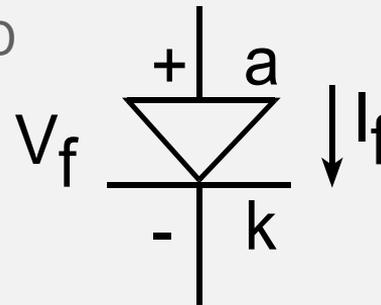


LED Interfacing

Resistor

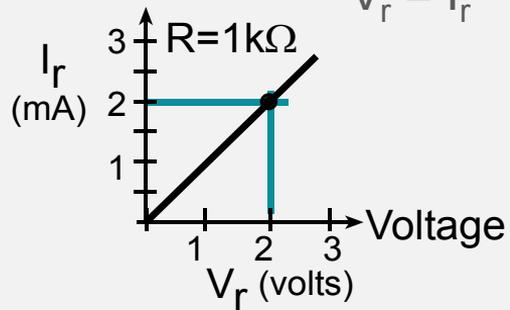


LED

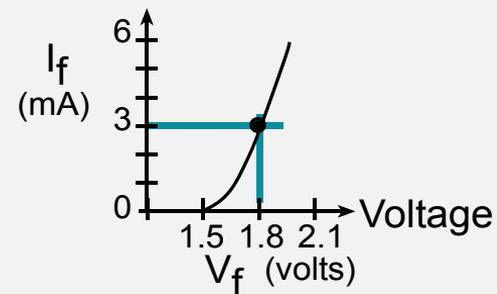


Current

$$V_r = I_r * R$$



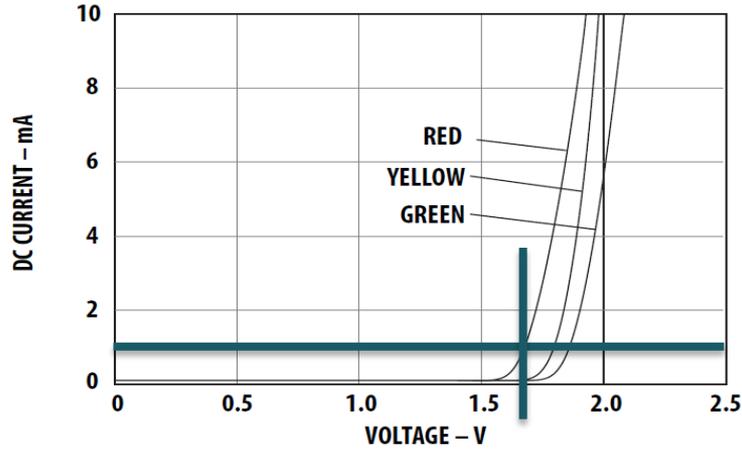
Current





LED Interfacing

LED current vs voltage



1 mA, 1.6V

Brightness = power = $V \cdot I$

anode (+)

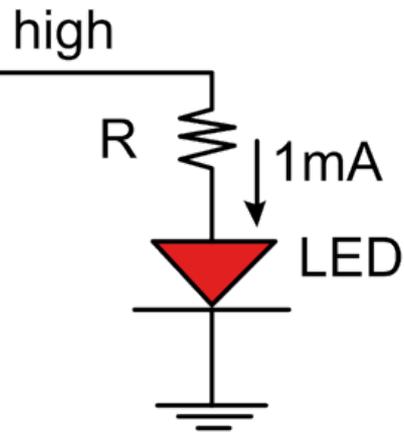
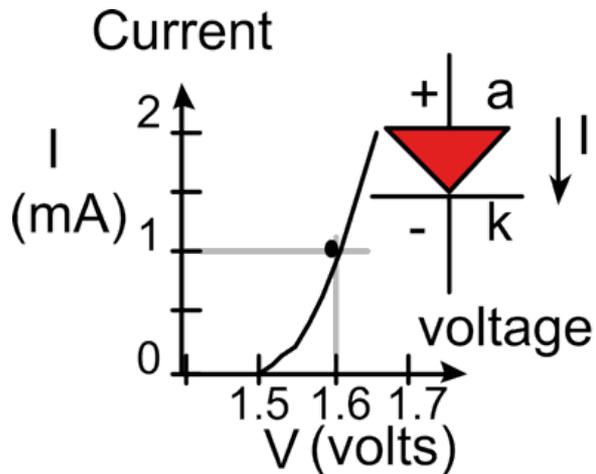


cathode (-)

“big voltage connects to big pin”



LED Interfacing ($I < 6 \text{ mA}$), Positive Logic



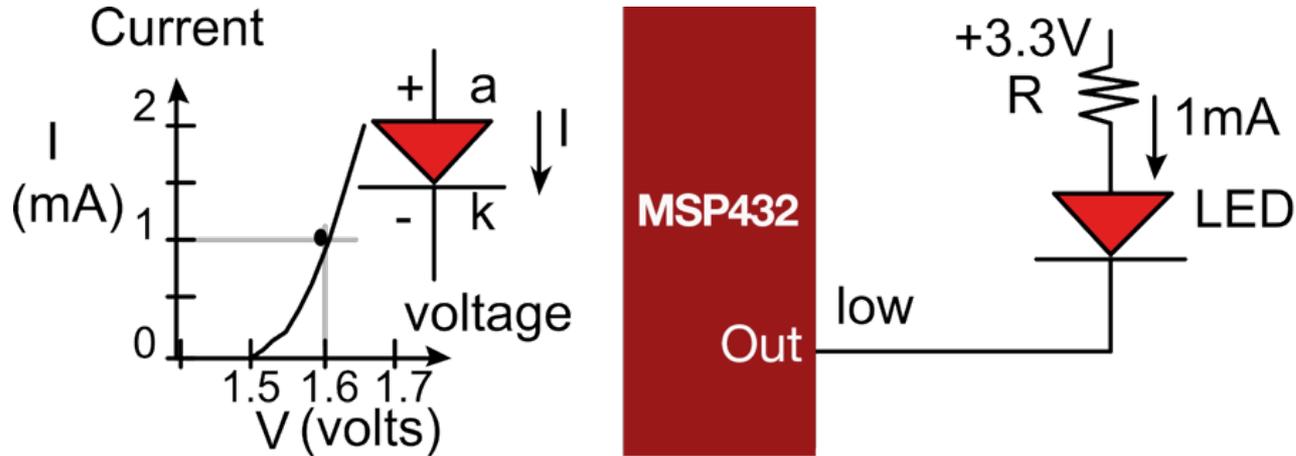
$$R = (3.3\text{V} - 1.6)/0.001\text{A} = 1.7 \text{ k}\Omega$$

Standard R = 1.6 k Ω

Brightness = power = $V \cdot I$



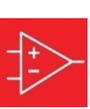
LED Interfacing ($I < 6 \text{ mA}$), Negative Logic



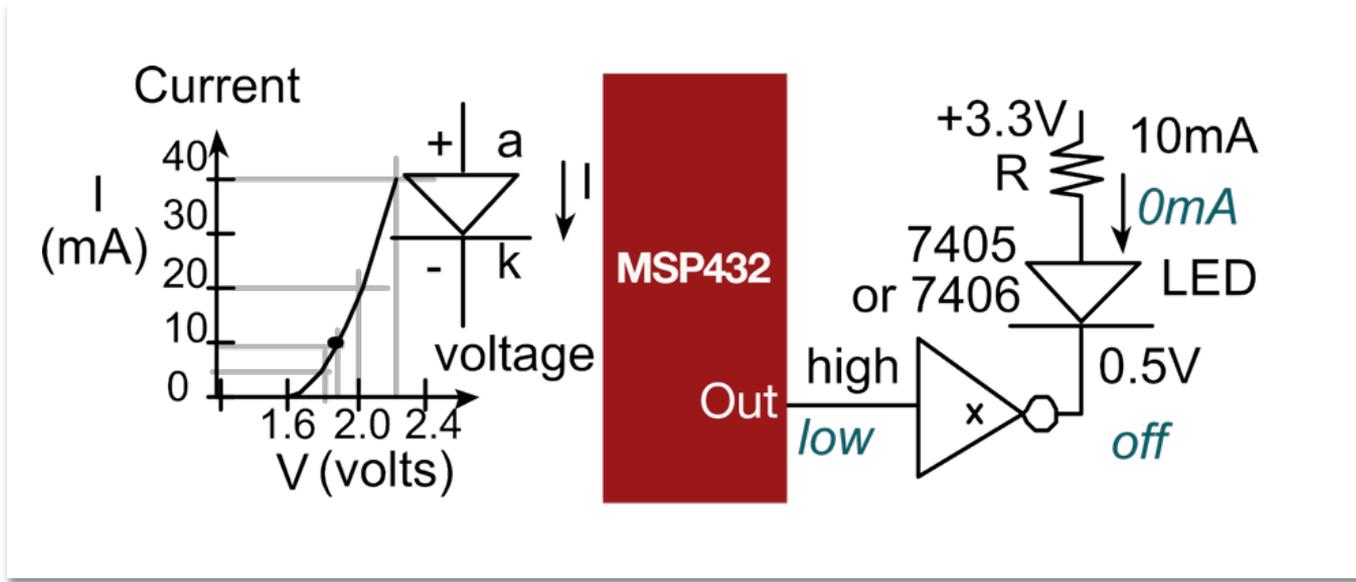
$$R = (3.3\text{V} - 1.6)/0.001\text{A} = 1.7 \text{ k}\Omega$$

Standard $R = 1.6 \text{ k}\Omega$

Brightness = power = $V \cdot I$



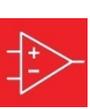
LED Interfacing ($I > 6 \text{ mA}$)



LED may contain several diodes in series

$$R = (3.3 - 1.8 - 0.5) / 0.01 = 100 \Omega$$

Brightness = power = $V \cdot I$



Software Driver (outputs)

Initialization (executed once at beginning)

1. Set *DIR* to 1 for output
2. Activate increased drive strength on output

Output to LED

1. Read from data output port
2. Modify bits as desired
3. Write to data output port

```
data = P2->OUT;  
data |= 0x01;  
P2->OUT = data;
```

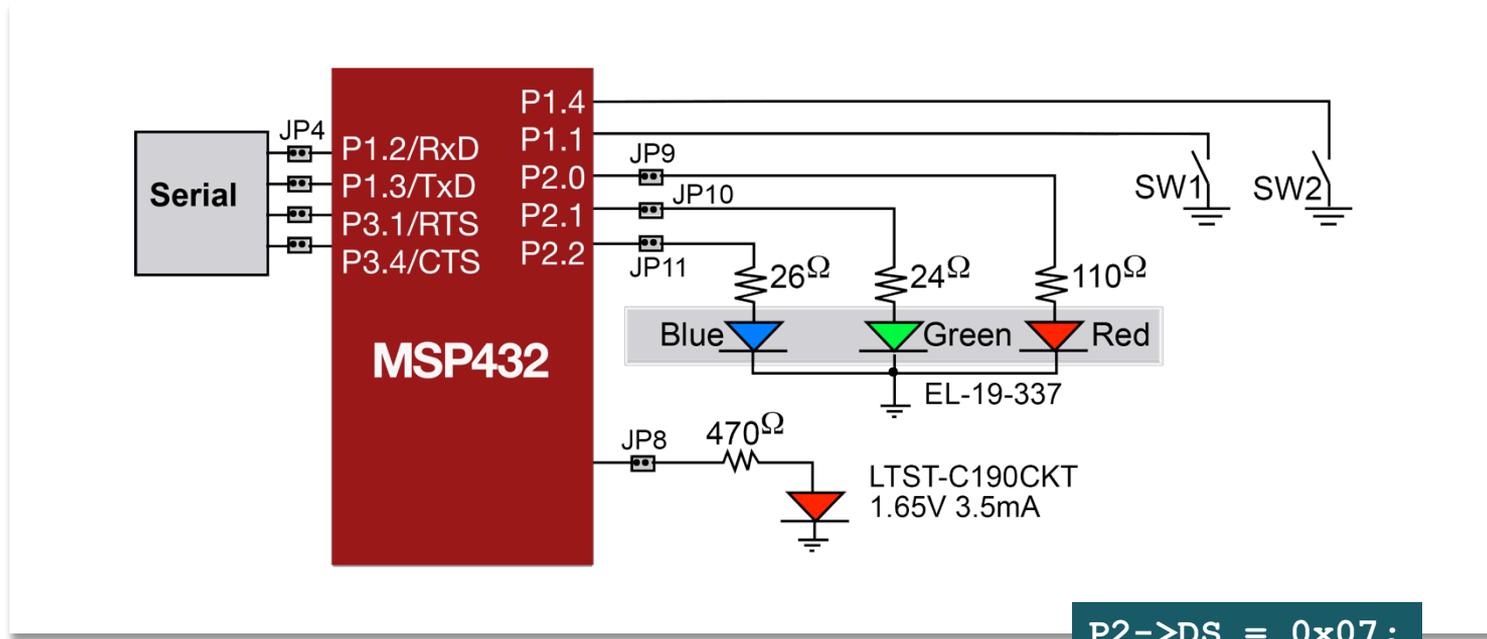
Set bit

```
data = P2->OUT;  
data = data&0xFE;  
P2->OUT = data;
```

Clear bit



LaunchPad Switches and LEDs



```
P2->DS = 0x07;
```

The LEDs are positive logic



Software Driver (simple, not friendly)

```
#include "msp.h"

void Port2_Init(void) {
    P2->DIR = 0x07;    // 1) make P2.2-P2.0 out
    P2->DS = 0x07;    // 2) activate increased drive strength
    P2->OUT = 0x00;    //    all LEDs off
}

void Port2_Output(uint8_t data) { // write P2.2-P2.0 outputs
    P2->OUT = data;
}
```

See [InputOutput_MSP432](#) example project



Software Driver (friendly)

```
void Port2_Init(void) {
    P2->DIR |= 0x07;    // 1) make P2.2-P2.0 out
    P2->DS  |= 0x07;    // 2) activate increased drive strength
    P2->OUT &= ~0x07;   //    all LEDs off
}

void Port2_Output(uint8_t data) { // write P2.2-P2.0 outputs
    P2->OUT = (P2->OUT&0xF8)|data;
}
```

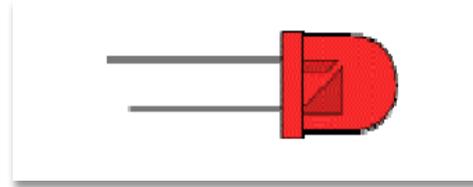
See [InputOutput_MSP432](#) example project



Application

Debugging

1. Control
2. Observability



LEDs provide

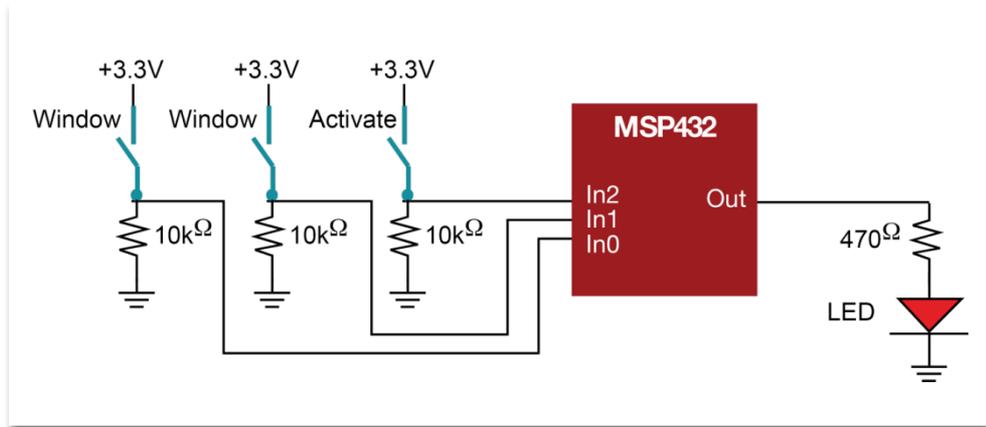
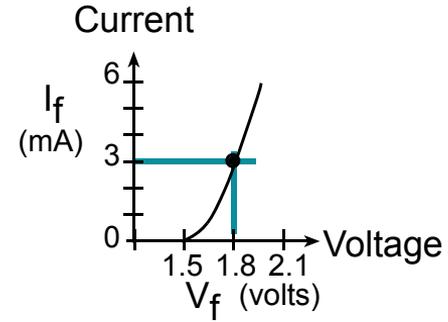
1. Diagnostic information for debugging (e.g., heartbeat)
2. Visualization of state (e.g., flashing rate signifies status)



Summary

- Positive and negative logic
- Ohm's Law for resistors
- LED nonlinear curve
- LED interface
 - Low current uses just a resistor
 - High current needs a driver
- Software driver
 - Initialization
 - Input/Output functions

$$V_r = I_r * R$$



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
Copyright © 2019, Texas Instruments Incorporated