Chapter 6: Supplemental Material

This chapter contains additional information about the electronics referenced in this course. It is intended for the people taking this course without a strong background in electronics.

Table of contents

Document conventions

Convention	Usage	Example
`Courier New`	Displays code and text commands	`CY_ISR_PROTO(MyISR);` `make build`
Italics	Displays file names and paths	*sourcefile.hex*
[**bracketed, bold**]	Displays keyboard commands in procedures	[**Enter**] or [**Ctrl**] [**C**]
Menu > Selection	Represents menu paths	**File > New Project > Clone**
Bold	Displays GUI commands, menu paths and selections, and icon names in procedures	Click the **Debugger** icon, and then click **Next**.

6.1 Internal Components

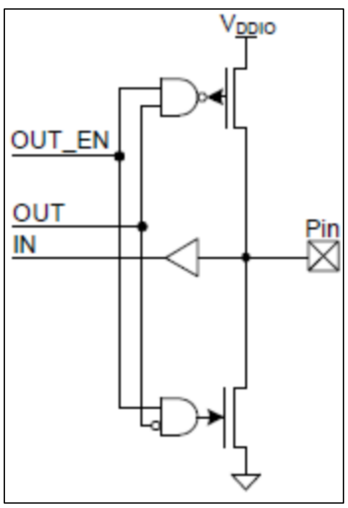
6.1.1 General Purpose Input Output (GPIO) (a.k.a. Pin)

GPIOs are the interface between the MCU and the outside world. They allow you to connect the device to just about any type of external component, be it analog or digital. To enable this functionality, BTSDK device GPIOs have a set of configuration selections that can be specified using the pin configuration function. These selections allow the input buffer to be enabled/disabled, the output to be enabled/disabled, resistive pull-up/pull-down paths to be enabled/disabled, the output drive strength to be selected, and input hysteresis to be enabled/disabled.

The ouput configurations that the GPIOs can take on are the following:

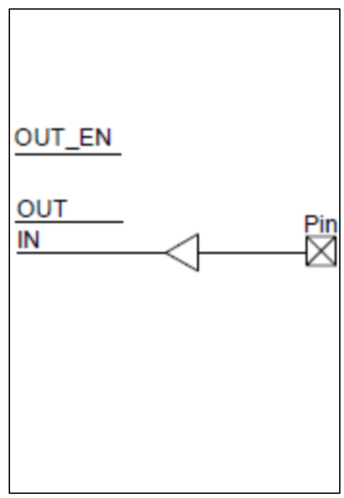
Note: *The triangles in these schematics represent input buffers. In each case, the input buffer can be enabled or disabled. When enabled, the input buffer can have hsysteresis enabled or disabled. Analog connections are taken directly from the pin.*

Output Enabled



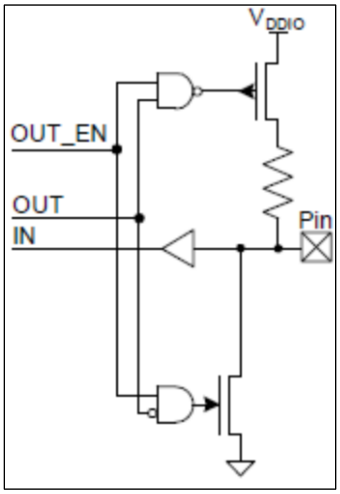
- This mode allows you to perform digital output operations. Although you can technically perform digital input operations in this mode it is **not** recommended.
- This mode is often used to drive LEDs.
- The drive strength can be configured to various levels

Output Disabled



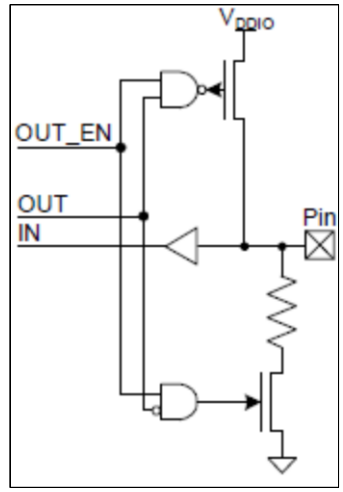
- This mode allows you to perform analog input and digital input operations. When inputting analog signals the input buffer is disabled.

Resistive Pull Up



- This mode allows you to perform both digital input and digital output operations. In this state, the pin is internally driven to a "1" through a resistor, but when it is internally driven to a "0" it is directly connected to ground. The resistive "1" value can be overpowered by a stronger "0" driver from another device on the board. This allows you to drive the pin to a "1" while simultaneously allowing external components to drive the pin to a "0". This is often used for wired-or configurations such as I^{2}C when there is no external pullup resitor on the board.
- This mode is often used for mechanical buttons that are active low. The output is set to a 1 so that it will pull the pin high when the button is not pressed.

Resistive Pull Down



- This mode allows you to perform both digital input and digital output operations. In this state, the pin is internally driven to a "0" through a resistor, but when it is internally driven to a "1" it is directly connected to ground. The resistrive "0" value can be overpowered by a stronger "1" driver from another device on the board. This allows you to drive the pin to a "0" while simultaneously allowing external components to drive the pin to a "1".

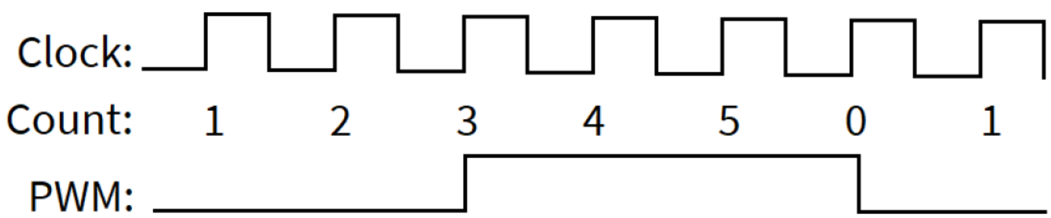
6.1.2 Pulse Width Modulator (PWM)

A PWM is simply a counter configured to count up from 0 to a specified period value with one or more outputs based on a compare value. A PWM can run continuously or require a trigger to begin running. Some common use cases for PWMs are:

- Creating arbitrary square wave outputs
- Driving an LED (changing the brightness)
- Driving Motors

A PWM outputs a "1" when the counter's current count value is >= its compare value and a "0" any other time. By changing the compare value relative to the counter's period value, you can change the percentage of time the output of the PWM is high – this is called the duty cycle. For example, a 50% duty would mean that the output is high half the time and low half the time. To accomplish this, you would set the compare value to be ½ of the period value. The speed of a PWM (the rate at which its counter runs) is based off of the PWM's clock signal.

Consider the following example in which a PWM has a period value of 5, a compare value of 3 and is configured to increment its count on the rising edge of its clock signal:



Note: A period of 5 means the full period is 6 clock cycles because the count starts at 0.

6.1.3 Analog to Digital Converter (ADC)

An ADC is a hardware component that can read a voltage from an analog pin and convert it into a digital value which the MPU can interpret. The ADC reads the voltage on a specified analog pin relative to ground or some other reference value.

The number of bits in the result an ADC produces determines the resolution of the ADC. For example, the result from a 16-bit ADC will be in the range:

$$-2^{15} \rightarrow 2^{15} = -32768 \rightarrow +32768$$

The raw result produced by an ADC has the unit "counts". Each "count" represents a fraction of the total input voltage range. For example: if the input range is 0 to 3.6 V and the ADC has a 16-bit resolution, then each count is: $\frac{5\ V}{(2^{16}-1)} \approx$ 0.0763 mV/count, which is also 13.1 counts/mV.

There is a function provided in the BTSDK ADC API to convert the count to a voltage in mV, so you don't need to do the conversion manually.

The AIROC™ CYW920835 device used in this class has a single-ended 16-bit ADC with an input range of either 1.8V or 3.6V.

6.1.4 Serial communication

Serial communication is often used to send data from one device to another. In serial communication, data is sent through one or more physical wires directly connecting devices, continuously, one bit at a time (hence the name serial). Data can be transmitted one direction at a time (half-Duplex), or in both directions simultaneously (full-Duplex).

There are a very large number of serial communication standards and protocols, but some of the most popular in the world of embedded systems are: UART, and I^2C. The specifics of how each of these protocols works is outside of the scope of this course, but if you are interested some good resources to start with are:

- UART Description
- I2C Description

6.2 External Components

6.2.1 Light Emitting Diode (LED)

An LED is a device that glows when you pass electrical current through it. The brightness of the LED depends on the amount of current that passes through it. If you pass too much current through the LED it will blow up (think fire and smoke). In general, LEDs are connected in series with a resistor that limits the amount of current (remember Ohms law? V=IR… look at the schematics below).

You can vary the brightness of an LED by either controlling the input voltage (which limits the current) or by "blinking" the LED faster than the human eye can see and varying the duty-cycle (see PWM). When using an MCU, it is much more efficient to use the "blinking" method to vary the brightness because it can be done by varying the duty cycle of a PWM rather than requiring the generation of a controlled analog output voltage.

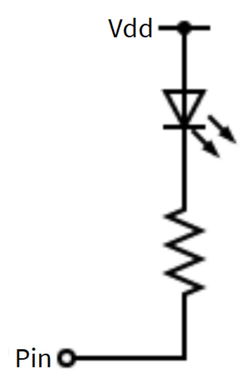
An LED is connected to an MCU in one of two ways:

- Active High (driving the LED with a "1" lights it up)
- Active Low (driving the LED with a "0" lights it up)

Active High

- When the MCU drives a "1", current flows out of the MCU, through the LED and into Vss (ground).
- When the MCU drives a "0", no current flows.

Active Low



- When the MCU drives a "0" the current flows out of Vdd, through the LED, and into the MCU.
- When the MCU drives a "1", no current flows.

6.2.2 Mechanical switch (push-button)

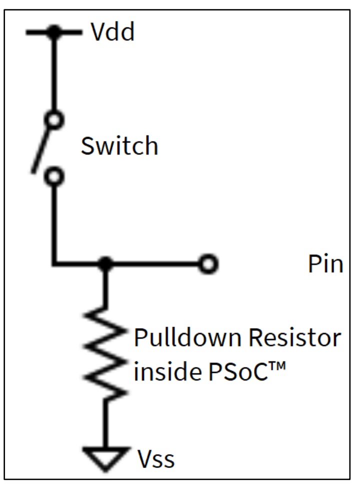
A mechanical switch is an electromechanical device (meaning that mechanical movements cause electrical actions) that electrically connects two terminals when it is in the closed position. A very common type of switch is a push-button. In this case, the terminals are connected when the button is pressed. When connecting a button to an MCU, the pin is typically configured using a resistor that pulls the pin to either Vdd or Vss. This is done so a separate resistor is not required on the board. On Infineon development kits, buttons are typically active low so the pin should be configured as an input with resistive pullup and the pin's initial state should be set to a 1 so that the pin will be pulled high when the button is not being pressed.

The circuit can be configured as:

- Active High (when the button is pressed the device reads "1")
- Active Low (when the button is pressed the device reads "0")

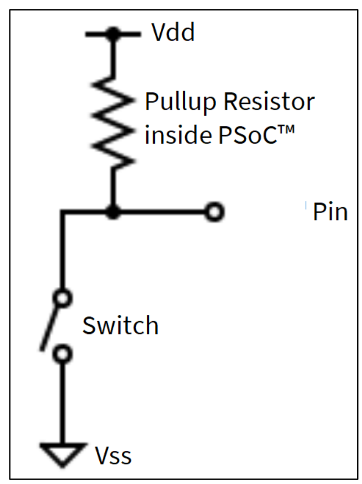
The active low case is much more common.

Active High



- When the switch is open (not pressed) the MCU will see Vss on the input and will read a "0".
- When the switch is closed (pressed) the MCU will see Vdd on the input and will read a "1".

Active Low

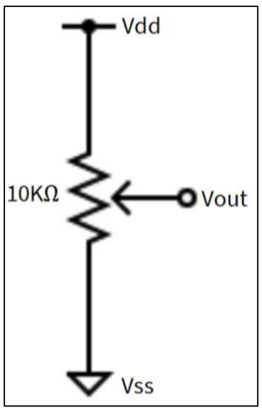


- When the switch is open (not pressed) the MCU will see Vdd on the input and will read a "1".
- When the switch is closed (pressed) the MCU will see Vss on the input and will read a "0".

6.2.3 Potentiometer (a.k.a. pot)

A pot is a 3-terminal electromechanical analog device. Two of its terminals normally connect to power and ground while the third terminal is used for output. The output terminal produces a voltage that varies between power and ground based on the position of the dial. Mechanically, a pot uses a sliding contact along a resistor to form an adjustable resistor voltage divider. A pot may be thought of as simply an analog voltage reference source.

The arrow in the pot schematic symbol represents the variable contact, controlled by turning the dial. As the contact slides toward the power rail (Vdd), the output voltage (Vout) rises higher. As the contact slides toward ground (Vss), Vout drops.

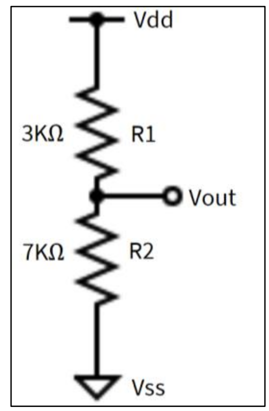


For example, suppose the dial is set so that the sliding contact is closer to the top power rail such that the top resistor is 3000 ohms and the bottom resistor is 7000 ohms.

In that case, the resistance between Vout and Vdd will be smaller than the resistance between Vout and Vss, as shown in the figure to the right. Remember the voltage divider equation?

$$Vout = Vdd\left(\frac{R2}{R1 + R2}\right) = 3.3\left(\frac{7000}{3000 + 7000}\right) = 2.31\ V$$

The output is 7/10ths of the power rail in this example.



Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Published by
Infineon Technologies AG
81726 Munich, Germany