|  |  |
| --- | --- |
| **Reading Time** | 20 mins |
| **Difficulty** | Intermediate |

|  |
| --- |
| **Sections** |
| Input and Output |
| Signal Types |
| Hardware Overview |
| Digital Input/Output |
| Analog Input |

Time Taken: 10 mins

What you’ll need: 1 brain

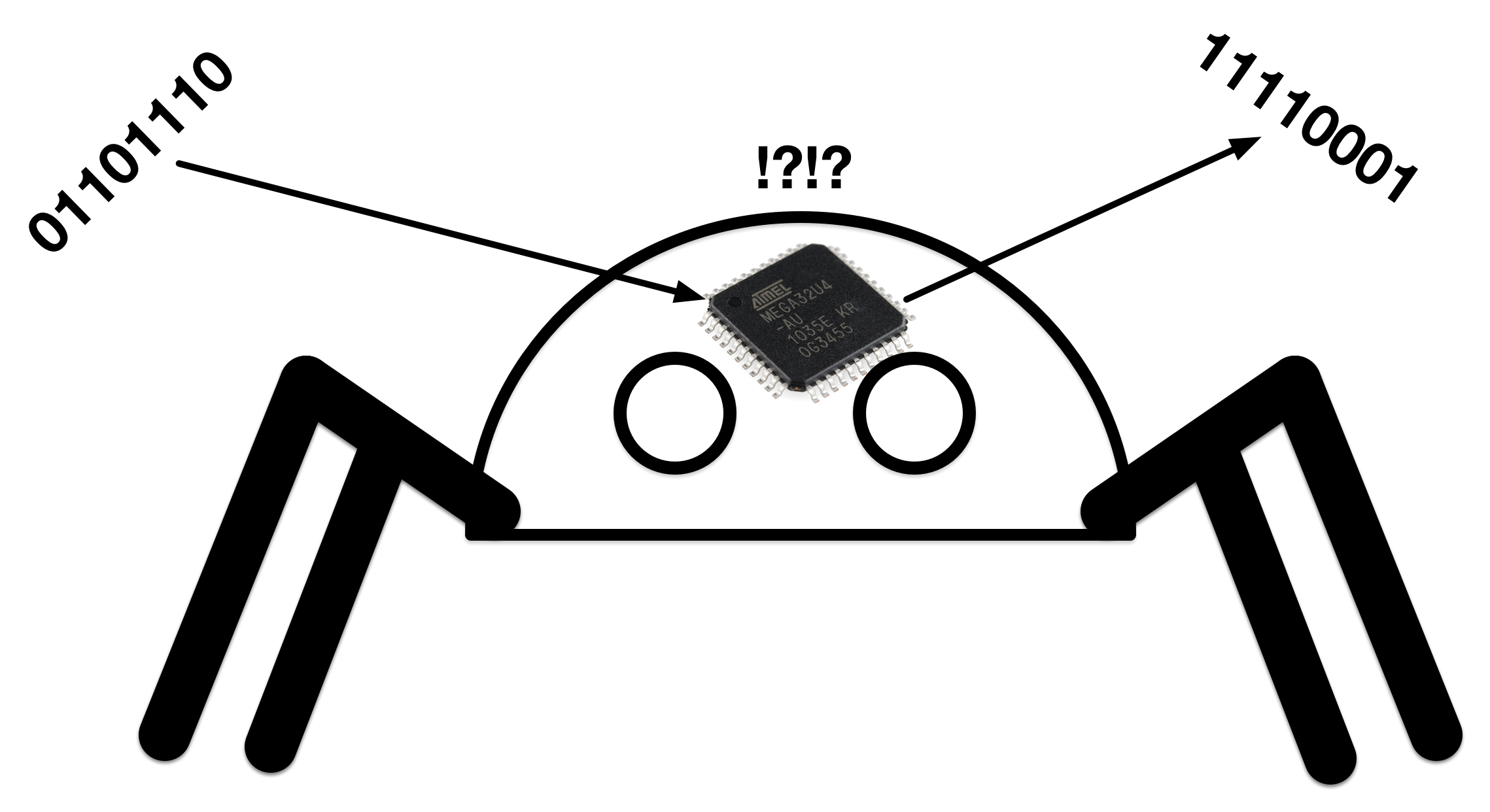
Difficulty: Easy

We’ll be covering:

1. Input and Output
2. Hardware Overview
3. Digital Input/Output
4. Analog Input

**Input and Output**

**Any robot must be interact with the outside world. That means the robot’s computer must be able to input and output information.**



**QuadBot uses a special type of computer called a microcontroller. A microcontroller isn’t as powerful as the computers in the device you’re reading this on, but it has some very useful features...**

1. **Input and Output: Microcontrollers have inbuilt features (called peripherals) to make input and output easy.**
2. **Lots of pins: Microcontrollers have a lot of pins for input and output.**
3. **Memory: Microcontrollers have inbuilt memory to store programs and data.**
4. **Single thread: Microcontrollers only runs one program at a time.**
5. **Embedded: Microcontrollers love to be embedded into larger systems, like robots.**
6. **Low power: Microcontroller are low power, perfect for battery powered robots.**
7. **Rugged: Microcontrollers are quite tolerant of voltages and electrical noise.**

**Let’s look at different types of signals…**

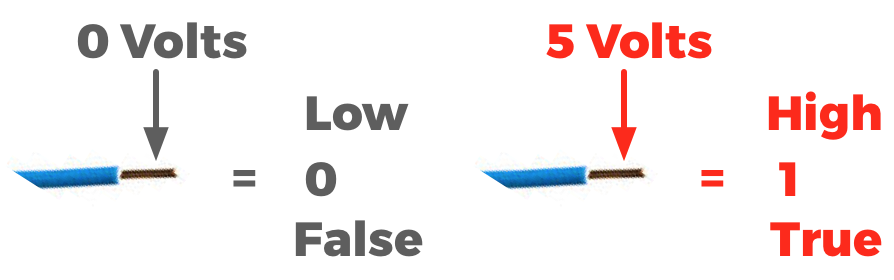
**Signal Types**

**Signals can be divided into one of two different types, digital and analog.**

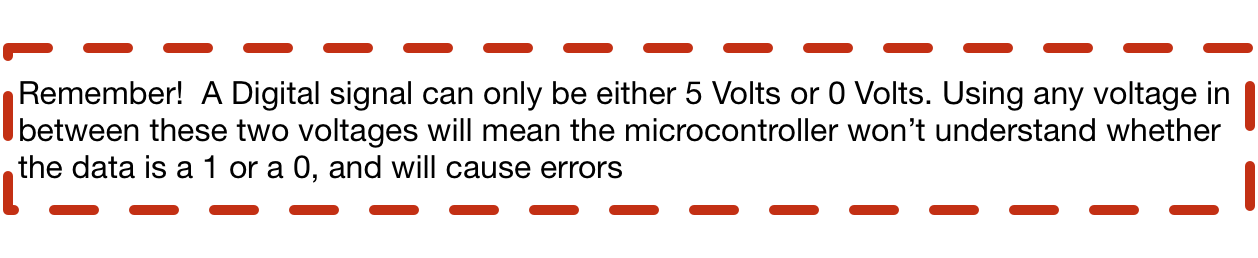
**Digital Signals**

**Digital Signals can be in only one of two states. On or Off. High or Low. True or False. Jekyll or Hyde (you get the idea).**

**Physically this it means either a high voltage or a low voltage. For QuadBot, a High signal means 5 volts on a wire, whereas a low one means 0 volts on a wire.**

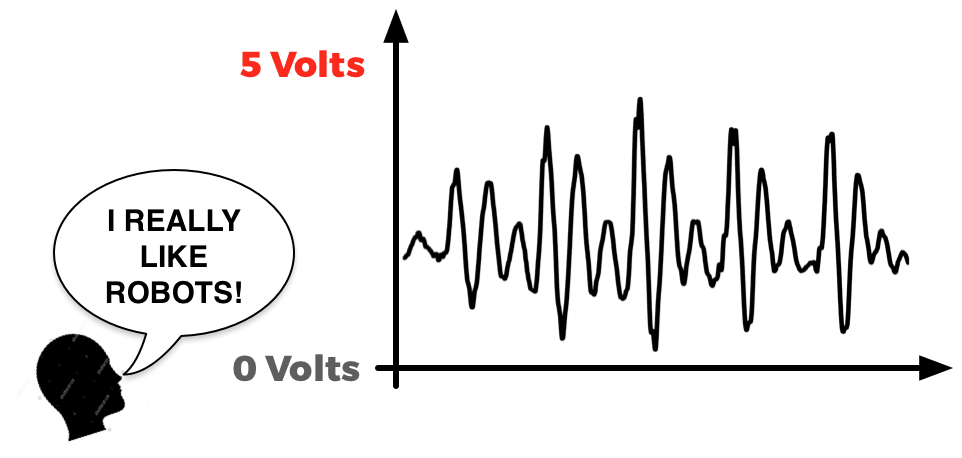


**This is how all computers work with information. Digital data is stored as voltages, either High or Low inside the computer. It’s a Boolean digIT so we call it a BIT.**



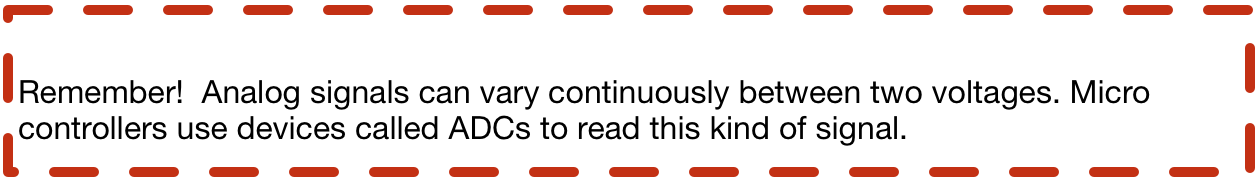
**Analog Signals**

**Analog signals are the opposite of digital signals. They can be any voltage between 0 and 5 volts. Analog is how the real world works for example the loudness of your voice isn’t just two values. If you used a microphone to convert this sound into a voltage, you’d see the voltage vary continuously between 0 volts and 5 volts.**



**Since microcontrollers are digital beasts, how can they understand an analog signal like that. Well they use a little device called an analog to digital converter (ADC) to read this data, and convert it into a digital value it can understand. And since microcontrollers are so awesome, these little ADCs are inside the microcontroller itself!**

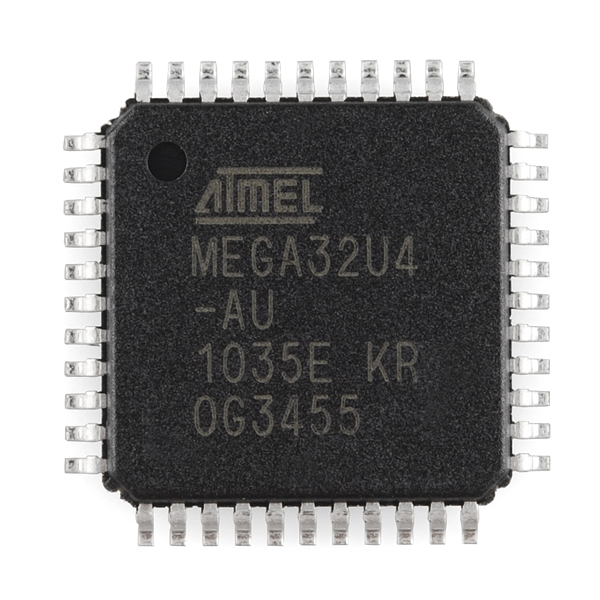
**Now let’s look into the microcontroller on QuadBot to understand the input and output capabilities we have.**



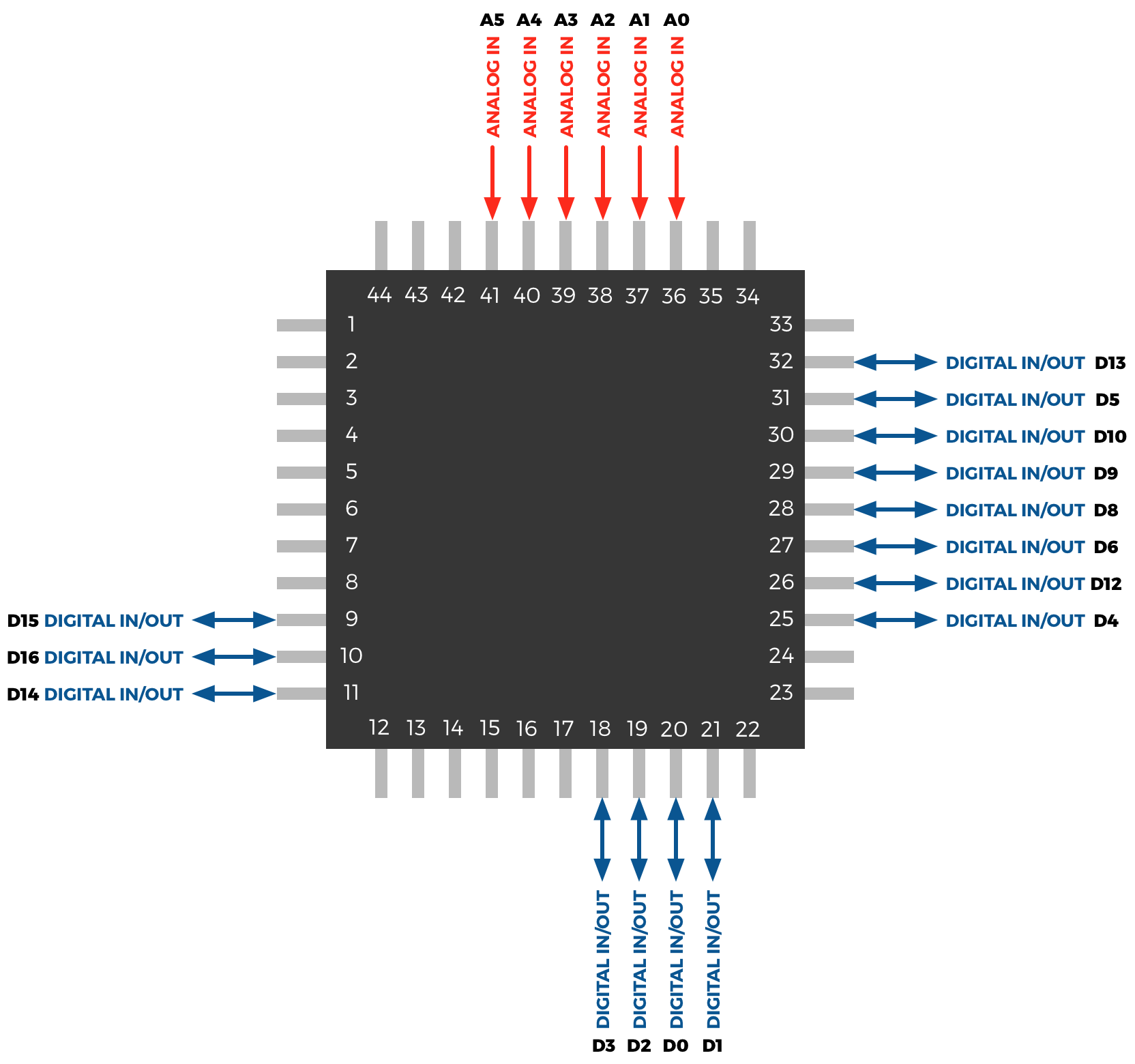
**Hardware Overview**

**The microcontroller**

**The microcontroller on QuadBot is called the atMEGA32u4.. Catchy right? It looks like this...**



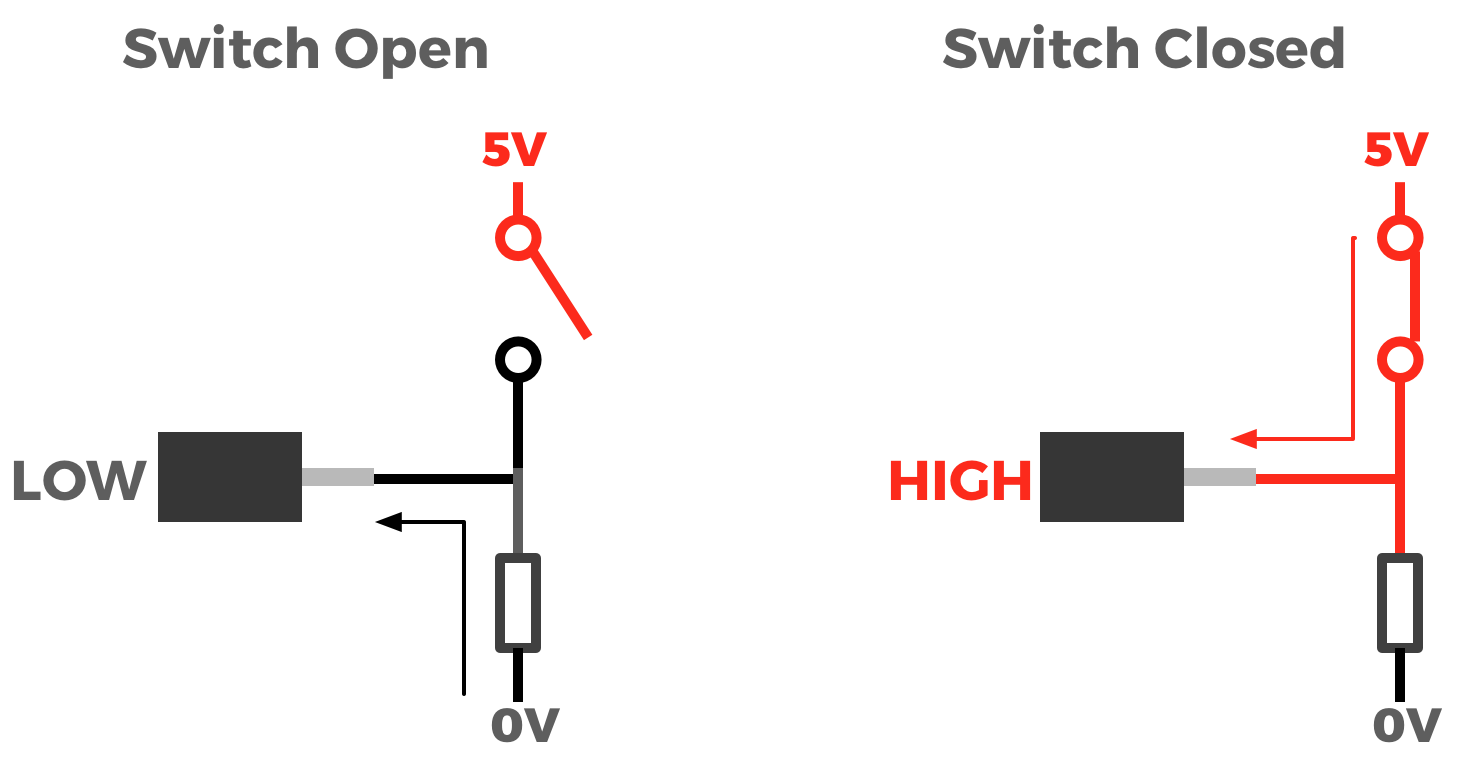
**It’s a black box with 44 metal pins. These pins allow it to communicate with the outside world, so let’s see what they do.**



**There’s 21 pins being used in total. The remaining pins do other functions that we won’t worry about in this tutorial (more on this later). There are two types.**

**Digital Input/Output**

**These are pins that the microcontroller can use to read digital signals from. Remember that digital signals can be in one of two states, so if we connected a switch to a digital input/output pin like this…**



**When the switch is open, we’d read LOW on this pin.**

**When the switch is closed, we’d read HIGH on this pin.**

**TRY THIS ON YOUR QUADBOT**

**To read this in our program, we can use the digitalRead function and store the value in a boolean variable like this...**

boolean switch = digitalRead(PIN);

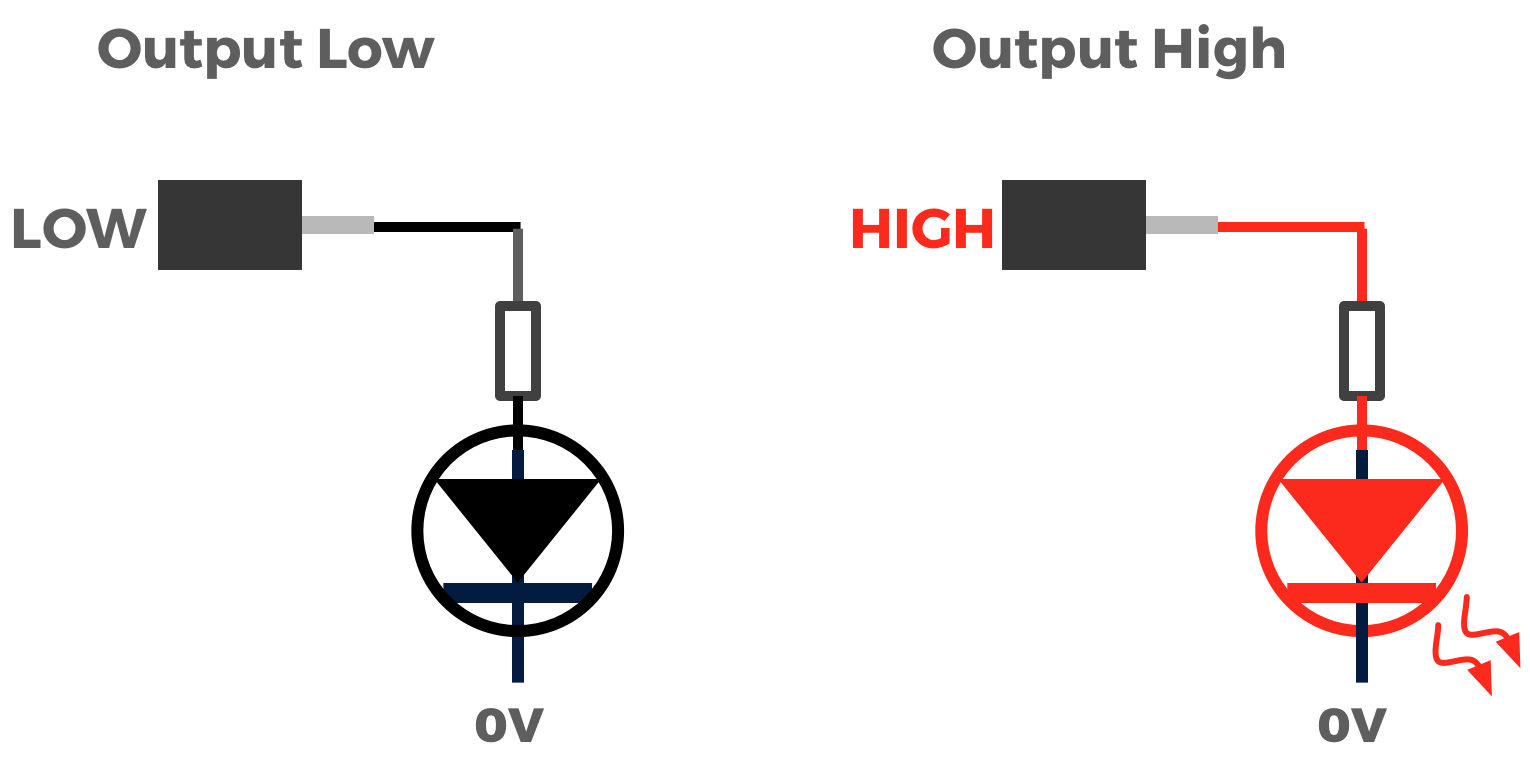
**Digital pins are special because they can also be used as either inputs or outputs. We have to choose whether we’d like a pin to be an input or an output in our program like this...**

pinMode(PIN, INPUT); //Sets PIN as an input

pinMode(PIN, OUTPUT); //Sets PIN as an output

**In the same program we can change a pin from an input to an output, but it cannot be an input and an output at the same time.**

**Let’s put an LED on a digital pin and see what happens.**



**We can set the output like this.**

digitalWrite(PIN, HIGH); //We set PIN to HIGH, turning on the LED

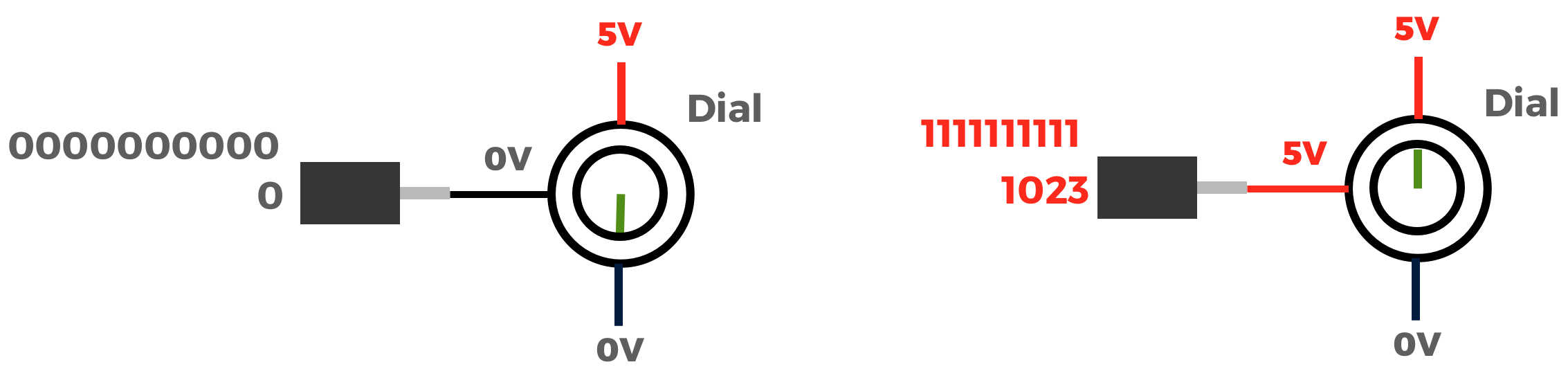
digitalWrite(PIN,LOW); //We set PIN to LOW, turning off the LED

**TRY THIS ON YOUR QUADBOT**

**Analog Input**

**Our microcontroller has six pins that can be used for analog input. Unlike the digital pins, these analog pins can only be used for input, they cannot output an analog voltage. So how does it work?**

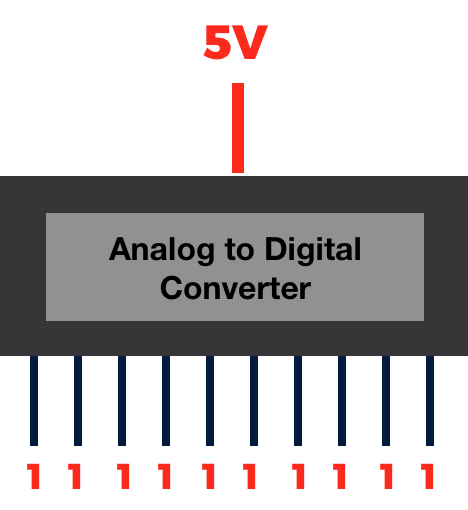
**Let’s look at the following circuit...**



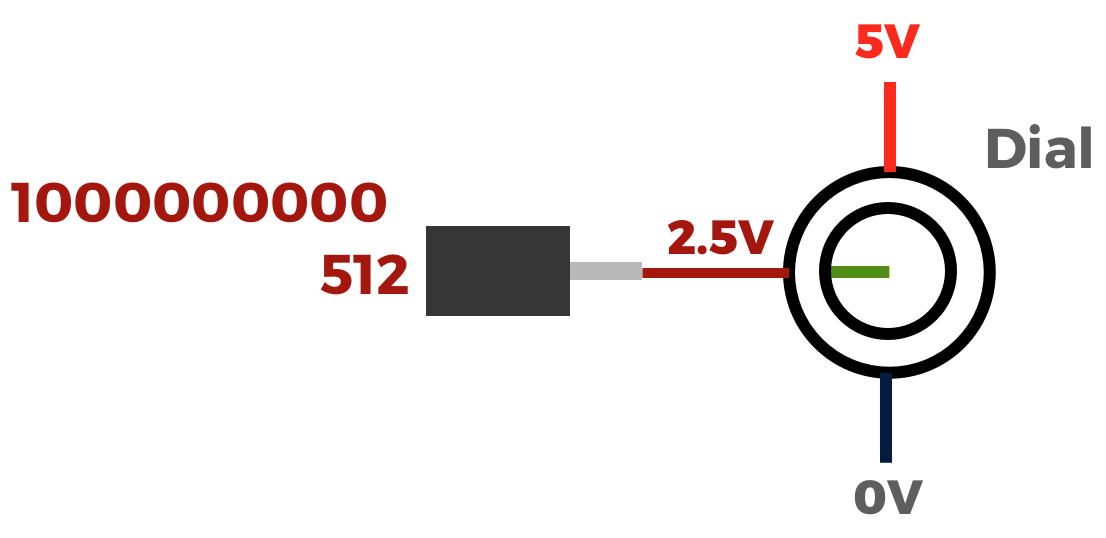
**We’ve connected a dial to an analog pin. That dial outputs a voltage between 0 and 5 volts into the analog pin.**

**When the dial is in the lowest position, 0V is at the analog pin of the microcontroller. What happens now?**

**The microcontroller reads this voltage as a number between 0 and 1023. 0 means 0V, whereas 1023 means 5V. Why does the microcontroller use the number 1023 as the maximum?**



**The microcontroller uses 10 binary digits (Bits) to store the voltage value, and since the largest number that can be stored in 10 bits is 1023, this represents the maximum voltage. What happens when the dial is in the middle position...**



**2.5 volts is read by the microcontroller as the value of 512 (1000000000 in binary), since 512 is 1023 divided by 2 (rounded up).**

**To use this in your program...**

int dial = analogRead(PIN);

**TRY THIS ON YOUR QUADBOT**