2

TECHNICAL SPECIFICATIONS OF THE PARALLAX PROPELLER

The Propeller Experiment Controller (PEC) is designed around the capabilities of the Parallax Propeller microcontroller. In this chapter, we will provide an overview the Propeller's hardware. We will not provide a comprehensive description of the Propeller's features, instead we will offer an informative framework as a necessary foundation for later discussion of the PEC. The Propeller Datasheet (Parallax Semiconductor, 2012) and Propeller Manual (Martin, 2011) should remain the primary reference on the technical specifications of the Propeller. To make the most of this chapter, the reader should have a basic understanding of electronics. Many introductory resources are available online (see Appendix B), including Parallax's website.

Hardware Overview

The Parallax Propeller is an 80 MHz microcontroller with 32 I/O (input/output) pins and eight independent, 32-bit processors that runs at 3.3 volts of direct current (VDC). Microcontrollers, like the Propeller, are small, simple computers contained within a single chip. They are designed to interface with a variety of external devices, and often act as the control centers of everyday electronics such as remote controls, computer mice, microwaves, and thermostats. In behavioral research, microcontrollers may independently control an experimental apparatus, or indirectly relay instructions and information from a computer. Figure 2.1 shows an image of the Propeller. The entire microcontroller is about the size of a stamp, at 10 x 10 x 1.4 mm.

Figure 2.1: The Parallax Propeller microcontroller.

Although the Propeller can be purchased as a single chip, **Propeller development boards** are a good option for beginners, prototyping, and small production runs. In addition to the Propeller microcontroller, development boards typically contain an **EEPROM** (electrically erasable programmable read-only memory) that can be used to load a program to the Propeller, a **crystal oscillator** that enhances the Propeller's ability to accurately keep track of time, and connections and related circuitry to facilitate connecting the Propeller to a computer and power supply. Some development boards also have special connections to allow the Propeller to connect to computer monitors, keyboards, mice, and speakers. In order to get the most out of the PEC, an **SD (secure digital) card** is required. The Propeller ASC+ and the Propeller DNA development boards both offer the standard components and an SD card connection in a convenient package. Chapter 4 provides more information about obtaining a Propeller microcontroller and other required hardware.

Input/Output Pins

The Propeller has 32 input/output (I/O) pins. Each **I/O pin** can be connected to an external device that will act as an input or an output. **Input devices** are used by the Propeller to *detect* the world, and can record the activity of subjects, environmental variables, or commands from the user. **Output devices** are used by the Propeller to *affect* the world, and can present stimuli or consequences to subjects, control environmental variables, or present information to the user. For the sake of clarity, we will also classify external devices as digital, analog, or complex.

Digital devices only exist in an off or on state. The state of a digital device describes the absence or presence of voltage flowing through a circuit and can be represented in binary by a 0 or 1. Digital input devices often include equipment like levers, switches, and infrared beam-break sensors. Once a digital input device is connected to an I/O pin, the Propeller can determine the state of the input device by detecting if voltage is flowing through the input circuit and into the I/O pin. Digital output devices include lights, speakers, feeders, shock grids, heaters, fans, motors, and more. When a digital output device is connected to an I/O pin, the Propeller can send voltage through the I/O pin to turn that device off or on. I/O pins can also alternate between input and output states; this may affect the behavior of the device connected to the pin. For example, a piezo transducer used as an output transmits sound, while it detects sound and vibration when used as an input.

Analog devices are not limited to a simple off or on state. Analog information is represented by a range of numbers, such as 0–255, instead of 0 or 1. An analog input may record a variable such as *how much* voltage is flowing through a circuit, or *how much* infrared light is hitting a sensor. In behavioral research, analog input can be used to record variables such as muscle contractions on an electromyography (EMG) sensor, how hard a subject is pressing a lever, or how close a subject is to the end of a runway. Analog outputs may include precise control of motor speed, brightness of a light, or the voltage of a shock grid. Many analog devices can also be used digitally, although less information or control is available through digital means. All I/O pins on a Propeller are technically digital, however, several techniques allow the Propeller to be used with analog devices. Analog input can be achieved using an **analog-to digital-converter (ADC) chip,** while analog output can be emulated using **pulse-width modulation (PWM)** techniques. See Chapter 8 for examples of some of these techniques.

Complex devices are not easily categorized in the digital or analog category. Although these devices may use digital or analog techniques, the Propeller interfaces with each complex device in a unique manner, which may require use of multiple I/O pins. Some complex devices use **chip-to-chip communication protocols**, such as inter-integrated-circuit (I^2C), serial communication, or serial peripheral interface (SPI) protocols. Using such communication protocols, chips send digital signals in a rapid pattern that represents complex information. Communication protocols can be used by complex devices that function as either an input or an output. For example, an electronic thermometer may function as an input, sending information regarding temperature, in degrees Celsius, to the Propeller using an I^2C protocol. The Propeller then has access to temperature data that cannot be represented by a simple 0 or 1. Alternatively, an audio file player chip may act as an output device and play specific audio files as instructed by the Propeller via an SPI protocol. Some complex devices also use communication protocols to act as both input and output simultaneously. For example, a serial communication protocol can be used to send data back and forth between multiple Propellers or between a Propeller and a personal computer. In this example, a second Propeller or a computer is acting as a complex device interfacing with the first Propeller. Complex devices may also rely on other techniques, often unique to that type of device, to interface with a Propeller. Computer monitors, for example, require the Propeller to use **VGA (video graphics array)** or **composite video** methods to display text or images.

Most microcontrollers have **specialized pins** for each function. Some pins may operate only as digital pins, others only as analog, and others still only for specific communication protocols. The I/O pins on the Propeller are not specialized; any pin can be used for most purposes. This makes the Propeller more flexible than many microcontrollers. There are, however, a few instances where some pins have unique functions. When the Propeller initially turns on, four pins have a momentary special purpose. Pins 28 and 29 use an I^2C protocol to load a program from memory, while pins 30 and 31 use a serial protocol to load a program from a computer. After boot-up, these pins can be used for any purpose, but this should be considered an intermediate technique. Video generation also has some requirements for specific pins. Composite video, depending on type, requires up to four adjacent pins (e.g., 0–3, or 8–11) and VGA video requires one of four blocks of eight adjacent pins (0–7, 8–15, 16–23, or 24–31).

All I/O pins function at 3.3 VDC, regardless of the type of device to which they are connected. When used as an input, I/O pins will consider voltage below 1.6 VDC to be off, and any voltage above 1.6 VDC to be on. I/O pins should never be directly connected to voltage greater than 3.3 VDC. When used as an output, each I/O pin can provide 3.3 VDC at 40 mA to an output device. See Chapter 8 for more information on making connections to some commonly used devices.

Multicore Architecture

The Propeller is relatively unique among microcontrollers in that it has eight independent, 32-bit processors called **cogs.** Each cog is identical and has access to all 32 I/O pins and several other resources. On boot-up, the Propeller will begin running a program in the first cog (cog 0). Then, additional programs may be started and stopped in other cogs as needed. During operation, a central **hub** allows each cog to operate sequentially. In a sense, the selection of which of the eight cogs is active "rotates" around the central hub. This rotating theme leads to the name of the Propeller, cogs, and the Propeller's programing language, Spin.

The multicore architecture of the Propeller provides many interesting multi-tasking applications. A single cog can be dedicated to a complex task, like monitoring a user interface or recording data to an SD card, while another cog implements an experimental protocol. This multitasking approach makes some tasks that would be difficult for other microcontrollers both possible and easy to implement. This approach also greatly simplifies programing requirements for newer users. For example, an entire cog can be dedicated to a simple task, like blinking an **LED (light emitting diode)** at a fixed rate, while a second cog records lever-presses. Most microcontrollers would require a single program to conduct both tasks, but with eight cogs, there is often little reason to force a single cog to execute multiple tasks. Many other microcontrollers make use of **interrupts** to multitask. Interrupts cause a program to temporarily jump to a different section of code, then return to the original location once that section is complete. Although powerful when used correctly, interrupts have the weakness of suspending one aspect of a program to implement some other task and can cause errors unless carefully designed. The Propeller's multicore architecture removes the need for interrupts, and as such it lacks interrupts by design.

Each cog has access to several shared resources. This includes all access to all 32 I/O pins, 32 kB of **ROM** (read-only memory) used to store font characters and mathematics tables, 32 kB of **RAM** (random-access memory) used to store a program and related variables and data, and the system counter used for precise timing. All cogs also have several individual resources, including two counter modules used for many advanced techniques involving frequency generation and measurement, and one video generator module.

Software Peripherals

Microcontrollers are designed to interface with a wide variety of devices, many which have their own special requirements like chip-to-chip communication protocols, or video generation protocols. To deal with the requirements of external devices, many microcontrollers also have specialized hardware to communicate with these devices. This often includes I/O pins used exclusively to connect to one kind of device, or pins designed specifically for interrupts. For many applications, selecting the right microcontroller to interface with a device is an important decision.

This concern, however, is mostly irrelevant when using the Propeller as it is much more adaptable than most microcontrollers. The Propeller does not have specialized hardware to connect to specific devices, instead it relies on **software peripherals** that can be easily changed to fit the current requirements. In many cases, a software peripheral can run on a dedicated cog, essentially transforming that cog into an emulator of specialized hardware. For example, the Propeller does not have hardware connections specialized for I^2C protocol as do many microcontrollers. Instead, an I^2C program may operate in one cog to communicate with an assortment of I^2C devices, and then pass this information to another cog that implements an experimental protocol. Use of software peripherals is very common in applications of the Propeller. In fact, the PEC software uses two cogs as software peripherals, one cog is used exclusively to communicate with an SD card, and a second cog is used exclusively to keep track of time in the experiment. The remaining six cogs can be used freely to conduct an experiment. This degree of flexibility, without specialized hardware, is simply not possible with other microcontrollers.

Operating Procedures

The Propeller's hardware goes through a standard operating procedure when used. When a Propeller receives power, it begins the boot-up procedure. The Propeller first looks for a connection to host computer using pins 30 and 31. If a host is detected, it communicates with the host and can load a program from the host into RAM. If no host was detected, the Propeller communicates with an EEPROM using pins 28 and 29. If an EEPROM is found, it loads the program from EEPROM to RAM. The Propeller then enters run-time procedure. Pins 28, 29, 30, and 31 can now be freely used by any program. The Propeller will enter shut-down procedure if it stops receiving power, if the hardware reset pin is activated, or if the REBOOT instruction is used in a program. The Propeller will enter boot-up procedure again if it receives power. Any programs running from EEPROM can thus be reset by toggling the Propeller's power supply.