



The Big Idea:

The common television remote control encodes messages in infrared light. These messages can be received and decoded with an infrared sensor (Lesson 12) and an Arduino.™ Being able to decode these messages enables two possibilities. The first is that remote controls can be used to control devices, such as robots and musical instruments, operated by the Arduino.™ The second is that by understanding how messages are encoded, you can program an Arduino™ to transmit its own messages via an infrared headlight, as used in Lesson 13. This combination of being able to receive and decode infrared messages, along with the ability to encode and transmit messages, enables an entire class of Arduino™ devices, such as communicators and laser tag.

Background:

Infrared used in Lessons 12, 13, and 15 was limited to the illumination and detection of obstacles. No actual information is exchanged. But by turning the infrared light on and off in patterns, actual information can be conveyed. And this is precisely what entertainment remote controls do. Pressing a button on a remote control causes a message to be sent. This message has two parts—the address and the command (Figure 16-1).

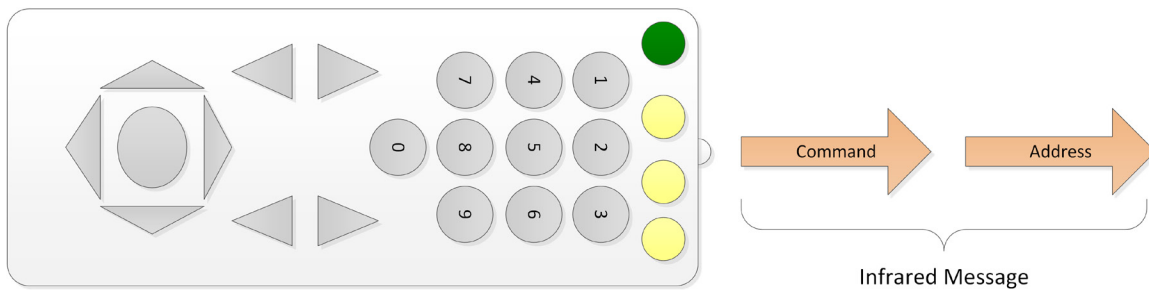


Figure 16-1. Infrared remote control and message

Table 16-1. Parts of an infrared message

Command	The action the device is to take. Examples are: increase volume, switch to a different channel, start playing.
Address	The identification of the device that is to respond to the command. Examples are: television, DVD player, satellite box.

The message is encoded by turning the infrared transmitter on and off in a pattern of pulses. Different manufacturers have different methods of encoding messages. In these lessons we use the method employed by the Sony Corporation and refer to it as the *Sony Infrared (IR) Protocol*.

Table 16-2. Vocabulary

Term	Definition
carrier signal	The base frequency of the infrared as it comes from the diode. In this class this frequency is 38,500 Hertz (cycles per second).
communications protocol	The set of rules used to encode and decode the information imposed with PWM on the carrier signal.
pulse width modulation (PWM)	The encoding of the carrier signal by turning it on and off in a pattern.
resting states	A period of no IR signal of a specific length that indicates the start of an encoded message.
Sony Infrared (IR) Protocol	The method of encoding infrared messages transmitted between remote controls and the Arduino™ that is used by Sony and other manufacturers and has become a de facto standard.

Description:

Recall that the output of the infrared sensor is HIGH when no infrared is detected and that this drops to LOW when infrared is detected. This is illustrated in Figure 16-2.

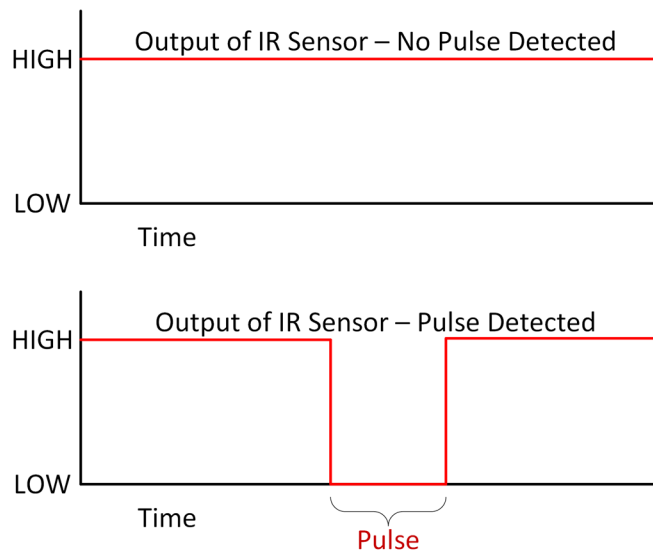


Figure 16-2. Pulse as it appears on the output of an infrared sensor

With this in mind, consider the format of a Sony-protocol infrared message as it appears on the output pin of an infrared sensor. This message has three parts, as shown in Table 16-3.

Table 16-3. The three parts of a Sony-protocol infrared message

Start	A pulse of unusual length that follows a period of no pulse. The start pulse indicates the beginning of a message.
Command	A set of seven pulses that indicates a pattern of seven ones and zeroes. This is a number in binary form. The value of that number indicates the action to be taken.
Address	A set of five pulses that indicates a pattern of five ones and zeroes. This, too, is a number in binary form. The value of this number identifies which device is to respond to the command.

Figure 16-3 is a diagram of a typical Sony-protocol infrared message. Notice that pulses are separated by so-called **resting states**. The duration of a data pulse indicates its value. A pulse duration of 0.6 milliseconds, for example, indicates that the bit's value is zero. A bit of one is indicated by a pulse duration of 1.2 milliseconds. The start pulse is 2.4 milliseconds.

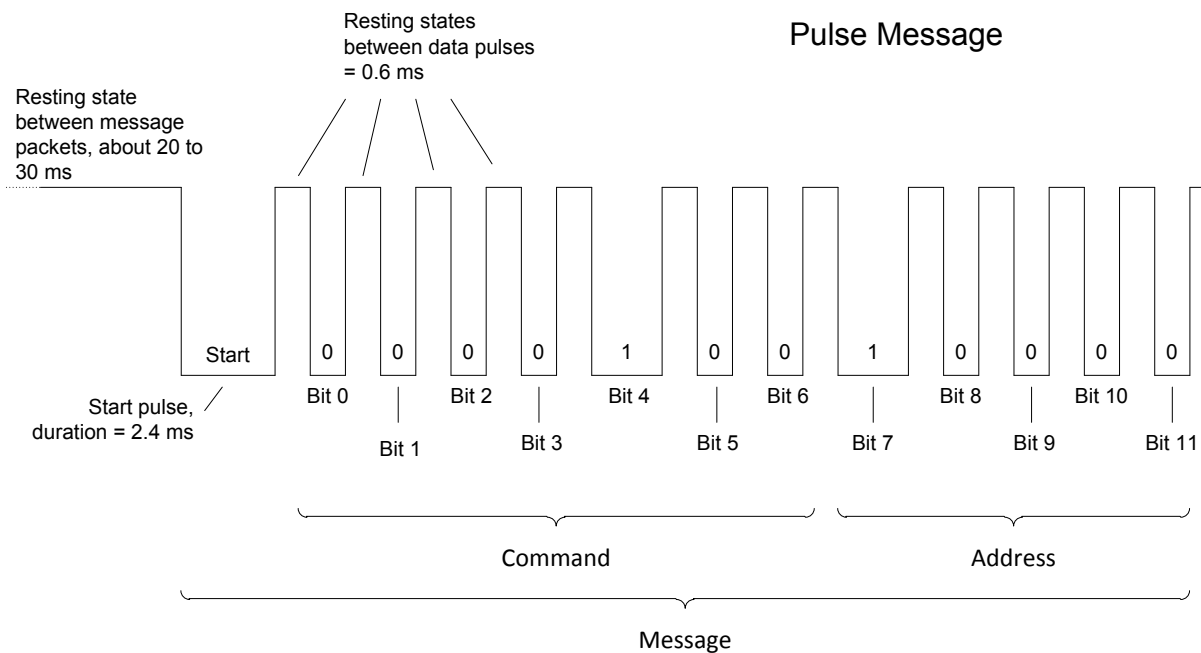


Figure 16-3. Format of a Sony-protocol infrared message

Detecting and decoding a Sony-protocol IR message is a four-step process, as shown in Table 16-4.

Table 16-4. Steps for detecting and decoding a Sony-formatted message

Step 1:	Wait for the beginning of a message. In this step the Arduino™ program waits for the output of the infrared sensor to drop to zero. If this doesn't happen within the time expected for a rest period, then we stop and return without doing anything more. There's no message. But, if the output does drop, then IR is detected.
Step 2:	Measure the duration of this first pulse. If it is not about 2.4 ms, then the pulse is not a Start pulse. Return without doing anything more. But if the duration is about 2.4 ms, then a start pulse has been detected. The next 12 pulses are the command and message.
Step 3:	Get the next 12 pulses and determine the command and address. If there are fewer than 12 detected pulses or if a pulse exceeds 1.2 ms or if a rest period is longer than 0.6 ms, then return without doing anything more.
Step 4:	If all 12 pulses are detected and are either 1.2 ms or 0.6 ms, then decode the message into its command and address; then return. Detection is a success.

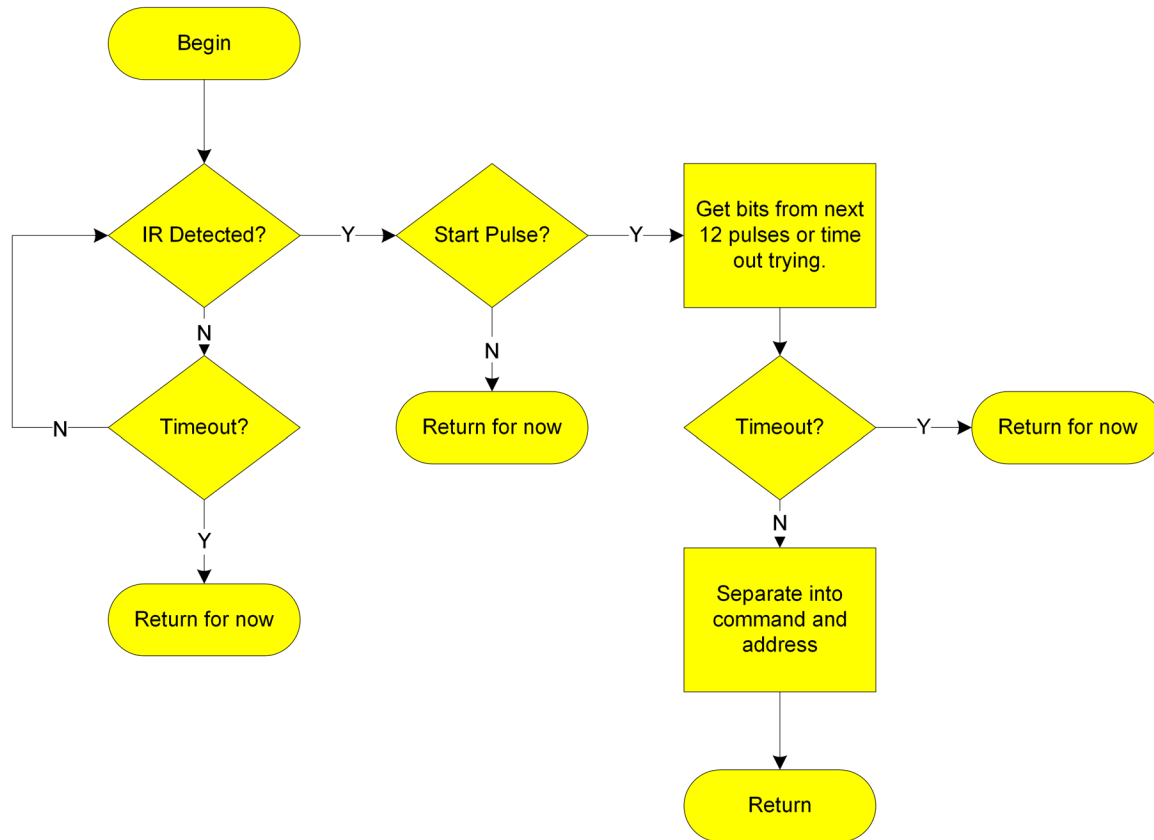





Figure 16-4. Flow chart of detecting and decoding Sony-protocol infrared message

Goals:

By the end of this lesson readers will:

1. Know that an ordinary television remote control uses infrared technology to send commands.
2. Know that each button on the remote sends different information identifying which button is being pushed. Further, this information is conveyed by turning the infrared light on and off in a pattern.
3. Know the definitions of:
 - a. Pulse width modulation (PWM)
 - b. Carrier signal
 - c. Communications protocol
 - d. Resting time between messages
4. Know that each time a button is pushed on a Sony remote control, an infrared message is transmitted and that this message has two components: the address of the device to be controlled and the actual command.
5. Extend your knowledge of the new C-language commands used with the Arduino.™
 - a. PIND: a special variable that reflects the input status of Arduino™ ports 0 through 7 in a single byte.
 - b. << : a special operation on a byte that shifts all the bits to the left.
 - c. DEC: a formatting command used with Serial.print and Serial.println to display binary data as decimal.
6. Be able to write a C-language program that can detect and decode a Sony-protocol message, yielding both the command and address.

Materials:

Quantity	Part	Image	Notes	Catalog Number
1	Arduino™ Uno		Single-board computer. This board is delicate and should be handled with care. When you are not using it, keep it in a box or plastic bag.	3102
1	USB Cable		This is a standard USB adapter cable with a flat connector on one end and a square connector on the other.	2301
1	Computer with at least one USB port and access to the Arduino™ website, http://www.arduino.cc .	---	The operating system of this computer must be Windows, Macintosh OS/X, or Linux.	---
1	Infrared sensor		3-pin, 38kHz.	1302

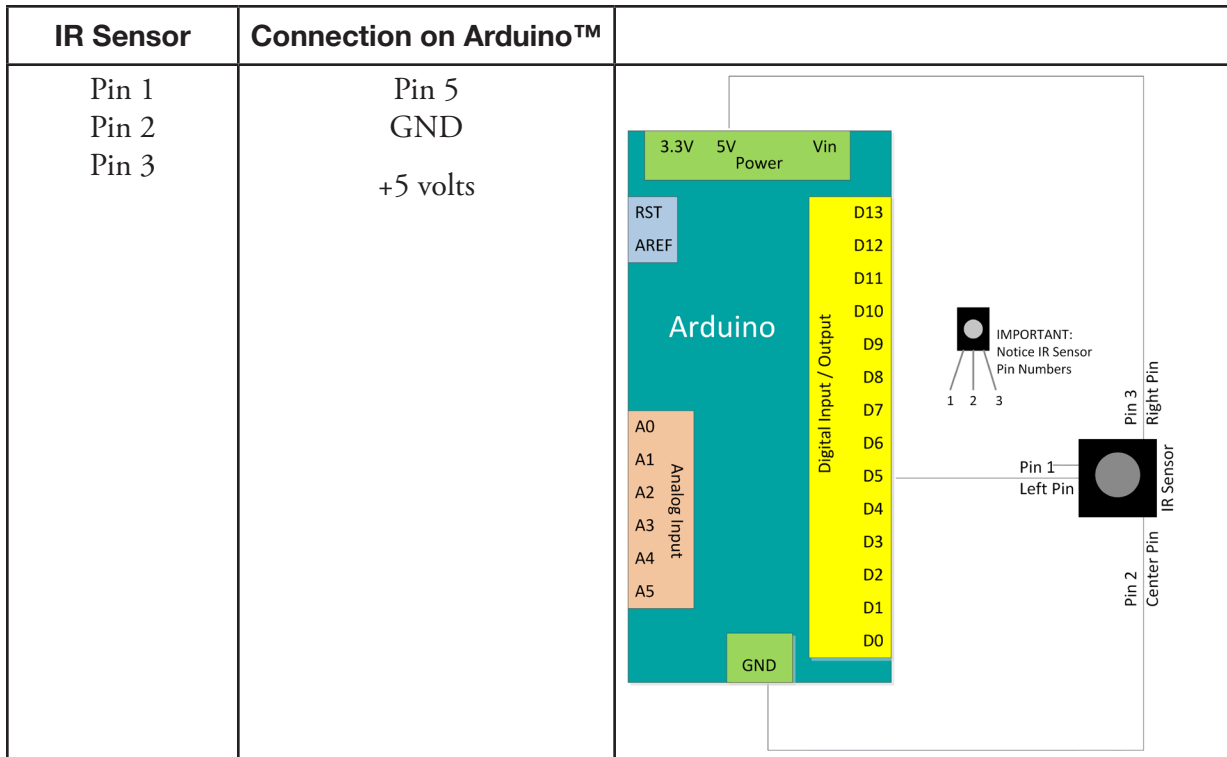


Figure 16-5. Schematic of connection between IR sensor and Arduino™

Procedure:

1. Connect an infrared sensor to an Arduino™ as shown in Figure 16-5.

 Important	Remember this is a schematic, not a pictorial. Pin 1 is NOT the center pin of the IR sensor!
----------------------	--

2. Create a new folder inside the Arduino™ sketch folder. Name it **Lesson16SonyIRProtocol**. THE NAME IS IMPORTANT, both the upper- and the lowercase letters.
 - a. Download the Arduino™ sketch Lesson16SonyIRProtocol.ino. Save it in the folder created in step 2. This sketch is available from Lesson 16 at LearnCSE.com.
3. Open the Arduino™ IDE. Then, from the menu, select File -> Sketchbook and open **Lesson16SonyIRProtocol.ino**.
4. Run the program to verify it works.
5. Complete Table 16-5, which lists for each button on the remote control the address and command code it transmits for each of the types of devices it can control.

	Television		Satellite/Cable		DVD		Other:	
Button	Ad- dress	Com- mand	Ad- dress	Com- mand	Ad- dress	Com- mand	Ad- dress	Com- mand

Exercises:

Exercise 16-1. Draw a flow chart of Lesson16SonyIRProtocol sketch

Look at the flow chart in this lesson; then look at the code in your sketch. Draw a flow chart that reflects how the sketch actually works. Is it different from the flow chart? If yes, in what ways?

Exercise 16-2. Modify PrintFull() to print device addresses and names of buttons

Modify the PrintFull() method to print in text the device for the address and the name of each button pushed.

