UNIVERSITY OF CALIFORNIA, DAVIS
Department of Electrical and Computer Engineering

EEC 172 Spring 2023

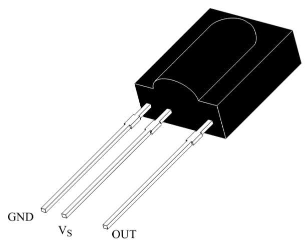
LAB 3: IR Remote Control Texting Over an Asynchronous Serial (UART) Link

Verification Due: May (3th MW or 4th TR) by the end of your lab section
Report Due: May (10th MW or 11th TR) in your lab section

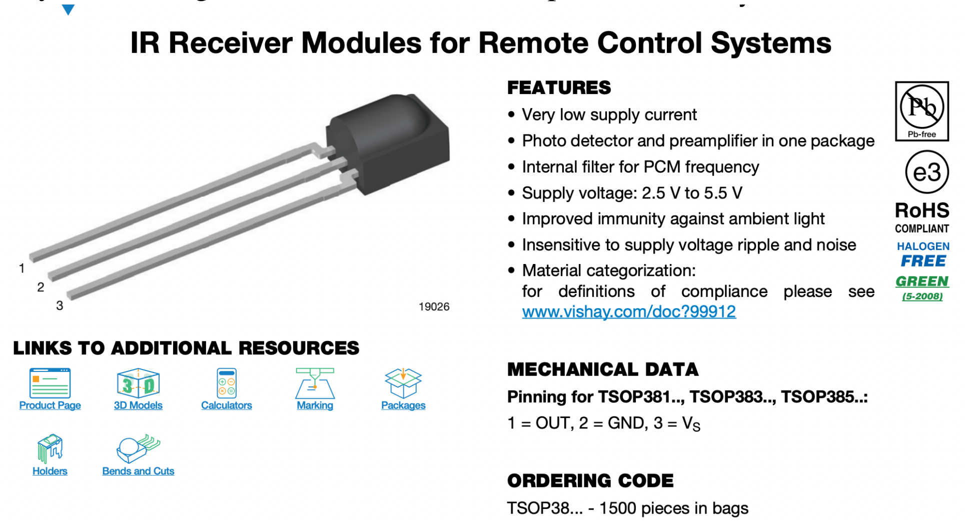
Objective: In this lab, you will use your Saleae logic analyzer and an IR receiver module to characterize the transmissions for buttons from an AT&T IR remote control for a specific TV. Each group will be assigned a unique TV code so that each group will be decoding different waveforms. You will then interface the CC3200 Launchpad to the IR receiver module and write a program that uses interrupts to monitor the signal from the IR receiver module. By analyzing the series of pulses, your program will be able to determine which button was pressed on the IR remote control programmed for your particular TV code. You will then use your IR remote control to compose text messages using the multi-tap text entry system (https://en.wikipedia.org/wiki/Multi-tap) and send text messages back and forth between two CC3200 LaunchPad boards over an asynchronous serial (UART) communication channel.

New Equipment Needed Per Group

2	AT&T S10-S3 Remote Control
2	Vishay TSOP31336 or 1236 or 31236 or TSOP38336
	(Datasheet in Technical Documents folder on Canvas - Pin-out below taken from datasheet)



If you are using TSOP38336 below is the pinout (**different from others**)



IR Receiver Modules for Remote Control Systems

FEATURES
- Very low supply current
- Photo detector and preamplifier in one package
- Internal filter for PCM frequency
- Supply voltage: 2.5 V to 5.5 V
- Improved immunity against ambient light
- Insensitive to supply voltage ripple and noise
- Material categorization: for definitions of compliance please see www.vishay.com/doc?99912

Pb-free
e3
RoHS COMPLIANT
HALOGEN FREE
GREEN (5-2008)

LINKS TO ADDITIONAL RESOURCES

Product Page · 3D Models · Calculators · Marking · Packages · Holders · Bends and Cuts

MECHANICAL DATA

Pinning for TSOP381.., TSOP383.., TSOP385..:
1 = OUT, 2 = GND, 3 = V_S

ORDERING CODE

TSOP38... - 1500 pieces in bags

2	100Ω resistor
2	100μF capacitor

IR Receiver Module Interface

Figure below shows how to wire the Vishay TSOP31336 IR receiver module. It has a very simple interface – power (Vs), ground (GND) and output signal (OUT). R_1 and C_1 are recommended to help filter noise that might occur on the power source and to protect against electrical over-stress (EOS) from power supply variations. Vs can be set to +3.3V and you can use $R_1 = \sim100\Omega$ and $C_1 = \sim100\mu F$. Because the IR receiver draws very little current, you can power it directly from your processor board.

Before connecting the IR receiver to your processor, you should wire it up and connect the OUT signal to an oscilloscope or the Saleae logic probe (See Part I). You will first collect waveform data with the Saleae module, before you interface the device to your Launchpad. When you interface the IR receiver to your processor, you will connect the OUT signal to an available GPIO input signal.

You should read through the TSOP31336 IR Receiver Module datasheet to understand how the module works. As shown in the Block Diagram of the datasheet, the IR receiver has a band-pass filter, which has a center frequency of 36 kHz. The TSOP31336 is able to suppress most of the interference from fluorescent lights. By using an oscilloscope, you can check if there are any spurious pulses on the OUT pin due to the fluorescent lights when there is no input signal from the remote. The lights in lab typically do not generate noise on the device, but outside of lab you may need to shield your receiver from the lights. When in the lab around other groups, you may need to shield your receiver from other groups' IR remotes.

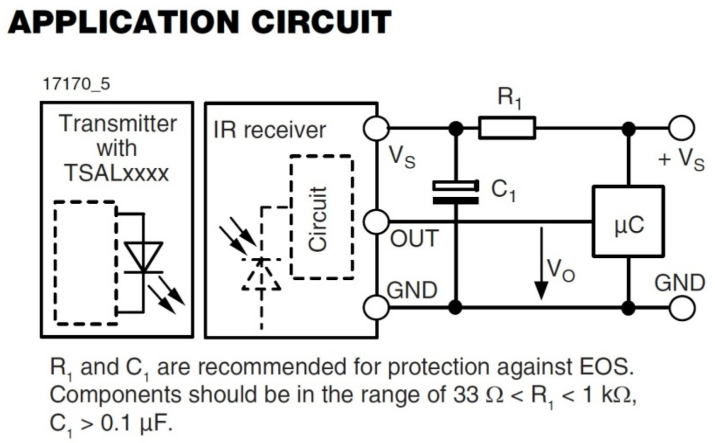

APPLICATION CIRCUIT
17170_5
Transmitter with TSALxxxx
IR receiver
Circuit
V_S
OUT
GND
R1
C1
μC
VO
+ VS
GND
R1 and C1 are recommended for protection against EOS. Components should be in the range of 33 Ω < R1 < 1 kΩ, C1 > 0.1 μF.

Figure 1. Application Circuit using the Vishay TSOP31336 IR Receiver Module (taken from datasheet)

Remote Control Configuration

- Your remote may have been previously configured so you should first reset the remote to factory defaults as follows:

    1. Press and hold the **AT&T** key along with the **OK** key. Hold both keys for one second, then release. All the mode keys should flash twice.
    2. Enter **900** using the numeric keypad. The AT&T key will give a long flash to confirm success.
    3. If the remote control times-out before you complete the procedure, you will need to start over.

- By default, the backlight is set to ON. You should disable backlighting to extend battery life. To toggle the backlight setting from ON to OFF (or vice versa), use the following steps.

    1. Press and hold the **AT&T** key along with the **OK** key. Hold both keys for one second, then release. All the mode keys should flash twice.
    2. Enter **991** using the numeric keypad. The AT&T key will give a long flash to confirm success.

- You will program a Device code for a specific TV, as assigned by your TA. A TV Device code is a 4-digit number that must start with "1". The IR transmission encoding for a specific TV varies by manufacturer and TV model so there are different Device codes for different TVs.

    1. Press and hold the **TV** key along with the **OK** key. Hold both keys for one second, then release. All the mode keys should flash twice to indicate that you are in the programming mode.
    2. Enter the 4-digit Device Code using the numeric keypad.
    3. The TV key will give a long flash to confirm success.

- Next you will configure the remote so that Channel and Volume commands always control one device, namely, the TV that you just specified. For channel control using the numeric keypad to be sent to your TV , perform the following steps:

    1. Press and hold the **AT&T** key along with the **OK** key. Hold both keys for one second, then release. All the mode keys should flash twice.
    2. Enter **966** using the numeric keypad. The AT&T key will flash twice to confirm success.
    3. Press the **TV** key. The TV key will give a long flash to confirm success.

    For volume control commands to be sent to your TV, perform the following steps:

    4. Press and hold the **AT&T** key along with the **OK** key. Hold both keys for one second, then release. All the mode keys should flash twice.
    5. Enter **955** using the numeric keypad. The AT&T key will flash twice to confirm success.
    6. Press the **TV** key. The TV key will give a long flash to confirm success.

Part I: Capturing and Characterizing IR Transmissions using a Saleae logic

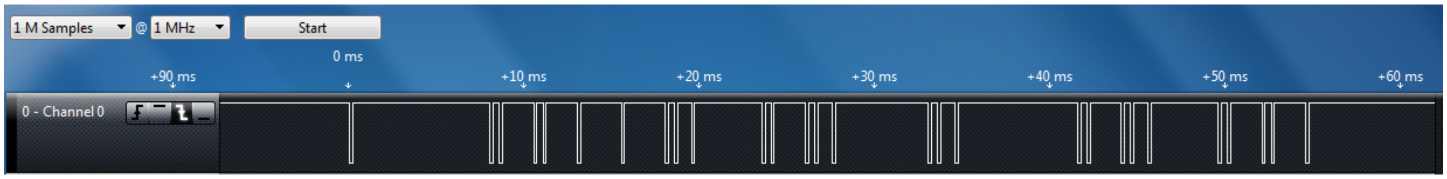
Each button on the remote is encoded with a different pattern of varying length pulses. See the document *Data Formats for IR Remote Control* posted in the Resources > Technical Documents folder in the class Canvas for some background on IR remote control transmission data formats and timing.

In Part I you will use the Saleae USB logic analyzer (http://www.saleae.com/logic) to capture and characterize the IR transmission format and timing for the remote control numeric buttons 0 through 9, the 'DELETE' button, and the 'ENTER' button. (For some TV codes, the 'ENTER' button may not be used, in which case use the 'MUTE' button.)

- When installing the wire harness into the logic analyzer pod, notice that the orientation is shown on the bottom of the pod. **Make sure that the ground symbol on the bottom of the Saleae module is aligned with the grey wire.** Black is data Channel 0, Brown is Channel 1 and so on (the colors follow the color code for resistor values).

- Our IR remote has a modulation carrier frequency of 36 kHz so a logic analyzer sampling frequency that is significantly above that (say 500 kHz) will yield accurate timing measurements. For example, you can set the sampling options to: 1 M Samples @ 500 KHz to capture 2 seconds of data or 1 M Samples @ 1 MHz to capture 1 second of data

- Once you have configured your IR remote, you can power the IR receiver module from a power supply set to +3.3V. The resistor and capacitor are recommended, but not essential. The OUT signal from the IR receiver would be connected to Channel 0 (Black wire) of the Saleae logic and the Ground probe (Gray wire) should be connected to the ground of the IR receiver. You will want to trigger on a falling edge on Channel 0. Press Start and then point the IR remote at the receiver and press a button. An example waveform is shown below.



- Use the logic analyzer to capture a transmission for each button in the target set. Use the Options > Save Screen Region feature to save a picture of the IR transmission code for each button. Another method is to use the PrintScreen key on your computer (Alt-PrintScreen) and then paste the clipboard into an application such as MS Paint for cropping. <u>You will need to include a printout of the waveform for each button for your lab report</u>. You should collect at least 2 waveforms for each button to make sure that the waveform is consistent and repeatable. For example, sometimes a waveform will change slightly on alternating button presses so that the receiver knows when one button press is released and another starts.

Use your set of timing waveforms to characterize the general format and general timing of your IR transmissions, and to determine the specific binary data that is sent for each button in the target set. If possible, use the IR protocol documents described above and other web resources to identify the protocol used for the IR transmission.

Some IR remote transmissions have an address and a data field. The address directs the data to a specific device (e.g., TV, DVD, VCR, etc.) and the data is essentially a key code which specifies which key (button) has been pressed. These key codes are sometimes called 'commands' because the key code is mapped inside the receiving device to a specific operation, e.g., Fast Forward.

When you press a button on the remote control unit and hold it down, the entire code may be repeated as long as the button is held down or the code may be sent once followed by a simple ‘repeat’ pattern that is common to all buttons. The receiving device ignores the repeated data, allowing a user to easily select Channel 3, for example, instead of accidentally getting Channel 33 (or 333, etc.) by pressing a button too long. Sometimes the code will have a bit that toggles each time a button is pressed so that the receiver can distinguish separate button presses. For our application repeated data can be programmatically ignored by discarding the repeated data fields.

<u>Part II: Decoding IR Transmissions / Application Program</u>

In this part, you will interface the microcontroller to the Vishay IR receiver module. The OUT signal of the TSOP31336 should be connected to an unused GPIO pin. Use the PIN MUX utility to configure the pin you choose as a GPIO input. Your software will use interrupts to detect the rising and/or falling edges of the IR remote input signal, and use one of the microcontroller's timers to determine the pulse widths. By analyzing the pattern of pulses, your program will determine which IR remote button was pressed.

You will write a program to decode the signal coming into the microcontroller to recognize when the user presses the following buttons:

Numeric keys: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

DELETE key (**or** LAST key)

ENTER key (**or** MUTE key)

Your system must distinguish these buttons from any other buttons on the IR remote. Your system does not need to identify other buttons, but it must recognize that they are not in the target button set. For example, pressing the LAST key should have no effect on your system (i.e. it should be ignored). (For some TV codes, the 'ENTER'/'DELETE' button may not be used, in which case use the 'MUTE'/'LAST' button.

Using the data you acquired in Part I, write a program that decodes and identifies the IR transmissions for each button in the target set (12 buttons) specified above and display them to a local terminal.

Demonstrate your work to your TA for verification.

Part II: Board to Board Texting Using Asynchronous Serial Communication (UART)

In this part, you will connect two CC3200 LaunchPads using asynchronous serial lines TX and RX of UART1 using interrupts. Your two CC3200 LaunchPads will each be interfaced to an Adafruit color OLED display.

Develop an application that uses your IR remote control to compose text messages using the multi-tap text entry system and send text messages back and forth (i.e. - bidirectional) between your two CC3200 LaunchPad boards over UART1. Both boards must be able to send messages to each other at the same time. The application must allow the user to input a text string via the IR and display the message on the local OLED.

In multi-tap texting, the intermediate characters should be displayed on the local OLED. For example, when typing the character 'z' the button '9' will be pressed 4 times, and the intermediate characters 'w', 'x', 'y' will appear at the current character position before 'z' appears. In order to type consecutive characters from the same button, a small delay will *confirm* the character. For example, to type 'bad', you will press the button '2' twice (to enter the character 'b'), wait a little, and then press the button '2' again (to enter the character 'a'). After entering 'a', you should be able to press the button '3' immediately after to enter 'd', namely you should not need to wait a delay cycle. The button '0' would serve as space key.

Pressing the button '1' must change the font color of the next input characters. The colors would be selected from a list of 5 preset options. The current font color should be shown on the OLED at all times. Then every time the button '1' is pressed the color would be changed to the next option on the list. You need to save the applied colors in a string.

Pressing the 'DELETE' key should delete the last character, to allow for correction of an errant character before a message is sent. Please note that in this lab we are not considering the numeric characters in the texting sequence. After the ENTER key is pressed the message and the color string must be transmitted to the remote board. The remote OLED must then display the received message with the correct font coloring.

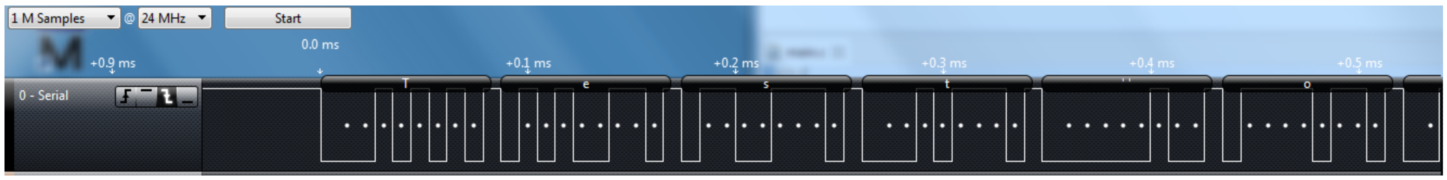
Incoming text messages should be displayed on the bottom half of the OLED. Messages under composition should be displayed on the top half of the display.

In previous quarters, we have noticed that students would simply 'use' InitTerm and not investigate the underlying UART api calls used to configure a UART port (which leads to difficulties in understanding how to open an additional UART port). There are several other methods that need to be called in order to open the UART port to allow interrupt-based communication (see http://software-dl.ti.com/ecs/cc31xx/APIs/public/cc32xx_peripherals/latest/html/index.html [**Note: Methods dealing with interrupts will often have a format of `*Int*(…)`, where * is a wildcard symbol, Int for Interrupt.]).

Here are two useful articles that give more insight on how interrupts work in general:
https://www.embeddedrelated.com/showarticle/469.php
https://www.embeddedrelated.com/showarticle/472.php

Demonstrate your work to your TA for verification.

For your lab report, capture a UART transmission from a message sent on UART1. Use the Async Serial protocol analyzer as shown in the example below.



Lab Report
At a minimum your lab report should include:

- Your verification sheet.
- A hard copy of your well-written, *well-commented* final program. Include your names in the header comments of the file containing the main program!
- A soft copy of your code uploaded to Canvas.
- Screen shots of the all IR transmission waveforms in your target set captured by your Saleae logic probe, labeled with the binary pattern that the transmission represents.
- An analysis and interpretation of the waveform data format used for your assigned TV set-up code, as well as which code you used.
- A printout of an UART waveform captured by your Saleae logic probe using the Async serial protocol analyzer.
- An explanation of how your program uses interrupts to decode the signals from the IR receiver module.
- A description of any noteworthy difficulties you encountered in constructing your solution.