Hacking a Classic Nintendo R.O.B. Robot

Thirty-three years ago in 1985 (which was considered post-North American video game crash of 1983 due to market saturation), Nintendo released a home-robot. R.O.B. (Robotic Operating Buddy) was an accessory to the Nintendo Entertainment System.

While R.O.B. was appealing, Nintendo only made a couple of games that used his capabilities (https://en.wikipedia.org/wiki/R.O.B.). However, R.O.B.'s success was limited, no more games were released, and R.O.B.s were relegated to closets, his gyro peripherals and claws scattered and lost to the robo-winds.

Revival Efforts

R.O.B.'s movements are commanded by the player using a series of precise flashes from a CRT analog television via a sensor in the robot's head. (It has no direct/wired connection to the NES itself.) But, using an NES on a modern digital LCD TV doesn't work! That's because R.O.B. uses a light sensor in his head which relies on specific flash timing that's a side-effect of how NTSC analog TV works.

Over the years, people have tried to recreate the R.O.B. Controls and there's been some success by Makers who have directly wired to the motor control board in R.O.B.'s base. We took this as a challenge. With the aid of some NES emulation detective work on the AtariAge forums along with Ladyada's NTSC-fu, we have recreated the light control sequence R.O.B. uses to move.

This tutorial will assist you in taking your dusty R.O.B. and making him a useful part of your life.

R.O.B. Buying Guide

If you want to go and buy a R.O.B. here are some hints:
1. You don't need an NES console or game cartridge. R.O.B. can work alone.
2. You don't need the spinners or other peripherals. The gray "claws" really are not required either.
3. You do want the battery cover for the bottom unless you plan to modify the power in some way.
4. You do want to ask the owner if R.O.B. works with the batteries in and if any gears may be stripped, as this will affect the price you pay.
5. Gear issues can often be fixed, but it may be worth it to buy a R.O.B. that seems in working order.
6. You don't need to buy one of the pristine bundles with or without original packaging unless you are a collector and know what you're spending money on. For this tutorial, only the basic R.O.B unit is needed.

A basic R.O.B. may go for US $50 to $100 - don't forget your local thrift shops!

Hacking Optical Control

R.O.B. uses a phototransistor in his left "eye" connected to a Sharp IR3T07 decoder chip. The IR3T07 is undocumented. But some creative folks at AtariAge.com reverse engineered old game cartridges to gather R.O.B.s control codes! Tursi found each command to R.O.B. consisted of 13 bits. The first five bits are an initialization string and are always the same: 00010. The next 8 bits are the command, coded as follows:

10111010 - Raises the body up
11110111 - Lowers the body down
10111010 - Turns the body left
11101010 - Turns the body right
10111110 - Closes the arms
11010110 - Opens the arms
11101011 - Turns the head LED on

Each bit is encoded as a green flash on the TV. We started with blinking a bright green LED with 60 Hz pulses (1/60 = 16.67 milliseconds) since NTSC has about 60 Hz framerate. No luck. Then we tried to use only the vertical blanking time within the 16.67 microseconds which is 1.5
milliseconds. So out of each 60th of a second, have the on/off bit active for 1.5 milliseconds, then off for the remaining 15.167 milliseconds.

This code was tested with an Adafruit 32u4 Feather and it worked! R.O.B responded to each of the commands!

The LED must be aimed at R.O.B.’s left eye (on the right as you look at him head on) and, unsurprisingly, brighter/narrower LEDs can be farther away than dimmer/diffused LEDs. Once we had some success, we tried other LED colors - turns out it doesn’t need to be green. White LEDs and infrared worked just as well. We’re not exactly sure why green was used in NES games; perhaps it was the brightest of the three phosphors?

Control Hardware

For our projects, we particularly like infrared because it isn’t noticeable to human vision. In the end, we then used an Adafruit Circuit Playground Express (CPX). The CPX is our all-in-one maker board. It has a transmit IR LED with a wide field of view and good transmit power (400mA+). The CPX shows up on your computer as a USB flash drive when connected via a power+data USB cable. Finally, it is programmable via CircuitPython, a language that is easy to use and code that can be changed quickly without installing a toolchain (like Arduino) and changes do not require recompile. Just copy a new code file onto CPX and it runs.

This makes the parts list very easy - you just need a Circuit Playground Express and a 3xAA battery back or USB cable.

Programming

The code is a relatively short CircuitPython program, and is available in the learn guide. You can copy it from the page or download it from the GitHub link. Save it onto a drive on your local computer as code.py


To load the code as-is, plug your Circuit Playground Board into your computer via the USB cable. You should see a new flash drive called CIRCUITPY in your list of available drives. In the off-chance your board does not have CircuitPython preloaded, follow the instructions in the guide to load the latest version. Drag a copy of code.py to the CIRCUITPY drive. The program will run immediately.

In the Python code, we set up a function IR_Command, which will do the infrared blinking. There’s definitions for the seven 8-bit codes plus the 5-bit init signal. That function performs the blinking as discussed above. If the bit is zero, the function delays 16.5 milliseconds. If the bit is a one, the LED is turned on for 1.5 milliseconds, turned off, and the program waits 15 milliseconds. 15 + 1.5 = 16.5ms. The while True: loop does the job of polling the Circuit Playground Express inputs and outputs commands appropriately.

If you have issues with R.O.B. not responding to the LED commands, carefully open the head up and remove the paper which restricts light to the left eye and blocks the right eye. This will allow more LED light in and make him easier to control.

It is best to experiment close to your computer with the USB cable still connected to the Circuit Playground Express. Use the serial console REPL to view debug information: printed info that the program has started and then (x, y, z) readings from the Circuit Playground Express accelerometer.

Hold your Circuit Playground Express about 12 to 18 inches from R.O.B.’s head with the IR LED not blocked by your finger and aimed at R.O.B.’s left eye (on your right). Don’t shake the Circuit Playground Express yet - hold it steady.

Now press the A button with a finger. R.O.B.’s arms should open wide (if closed). Press button B on the front of the Express and his arms should close. Hey, my robot works!

The other controls are a bit trickier - some practice will help. If you turn the Circuit Playground Express board clockwise, R.O.B. should twist one way. Turn it counterclockwise and he should move the opposite way. It may take some practice as the board is reading the changes in acceleration relative to the pull of the earth. Short, quick movements work best while keeping the IR LED aimed at R.O.B.’s head. Tricky but doable.

The final movement involves a quick up movement to have R.O.B. move his torso up. The silver USB connector should be moved up in a short quick movement (without twisting). R.O.B. should move the whole body/arm assembly up. A quick downward movement towards the ground should move the torso down. Again, keep the IR LED pointed at R.O.B.’s left eye. It may take some practice.

Once you have the basics down, you can unplug the USB cable and use the battery pack to make a portable solution. Go have fun with your robot!

Video:  https://www.youtube.com/watch?v=ffAuebA5WAr

Good night and good luck.