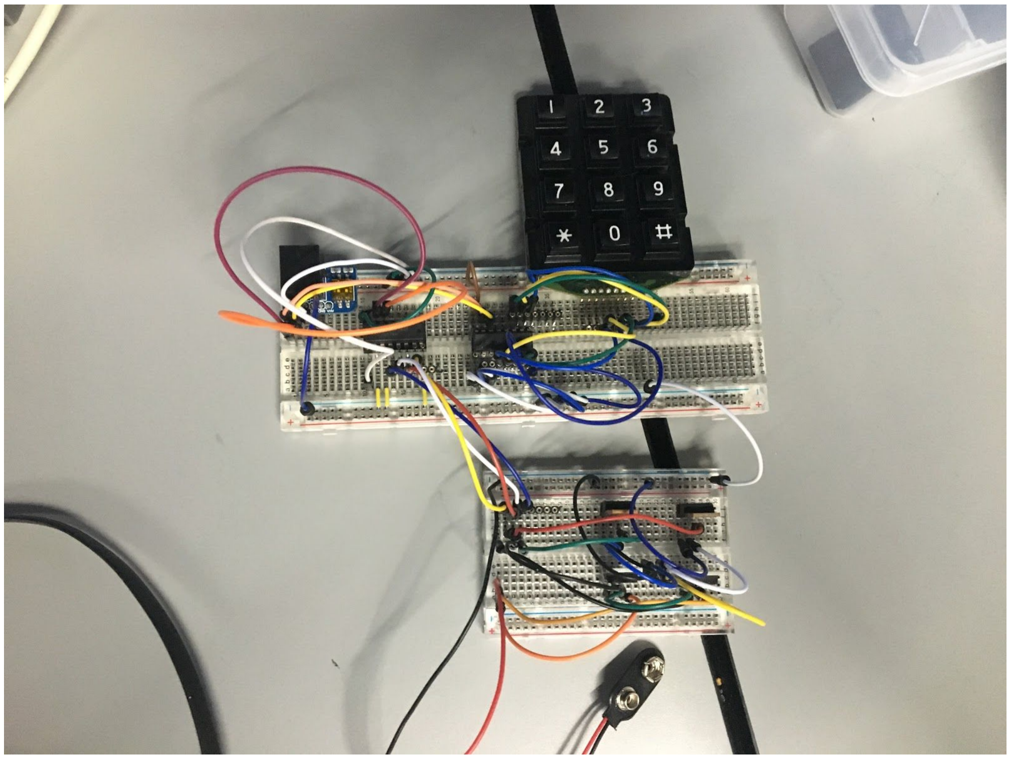
H-Bridge and Keypad Project

Logan T. Arias, Marco Arias

Keywords: 12-key keypad, PCF8574, NXT Brick, H-Bridge, Transistor

The photo depicts an H-bridge circuit, which is being controlled by a programmable numpad. This allows you which allows you to control the rotation and speed of a DC motor. This is particularly important in the field of controls, where using software and hardware to modulate the function of mechanical devices is extremely important. The big picture problem is understanding how to use both software and hardware to control things. Solving this partially or completely is important because the basis of automatic controls and robotics is in using these types of devices to do things. This tutorial shows you how to construct an H-Bridge, create a circuit, program a numpad using an NXT Brick, and takes approximately 1-2 hours to complete, depending on the person's past experience.



Motivation and Audience

This tutorial's motivation is to control a motor’s rotational direction. Readers of this tutorial assumes the reader has the following background and interests:

- ***Know how to operate the NXT software interface***
- ***Also know how to construct H-bridges and circuits, program in C***
- ***Knowledge of circuits and digital electronics is essential***
- ***Desire to further their knowledge of the field of automatic controls***
- ***In addition, would like to pursue a career in robotics***

Parts List and Sources

IRF510:
http://www.daslhub.org/unlv/courses/me425-625/lesson-L-robotInterfacing/lab/dataSheets/irf510.pdf

PCF8574:http://www.daslhub.org/unlv/courses/me425-625/lesson-K-robotCommunications/lab/dataSheets/PCF8574_PCF8574A.pdf

IRF9530: http://www.irf.com/product-info/datasheets/data/irf9530.pdf

Duracell: https://www.duracell.com/en-us/products/size/9v/

Keypad: https://www.sparkfun.com/products/14662

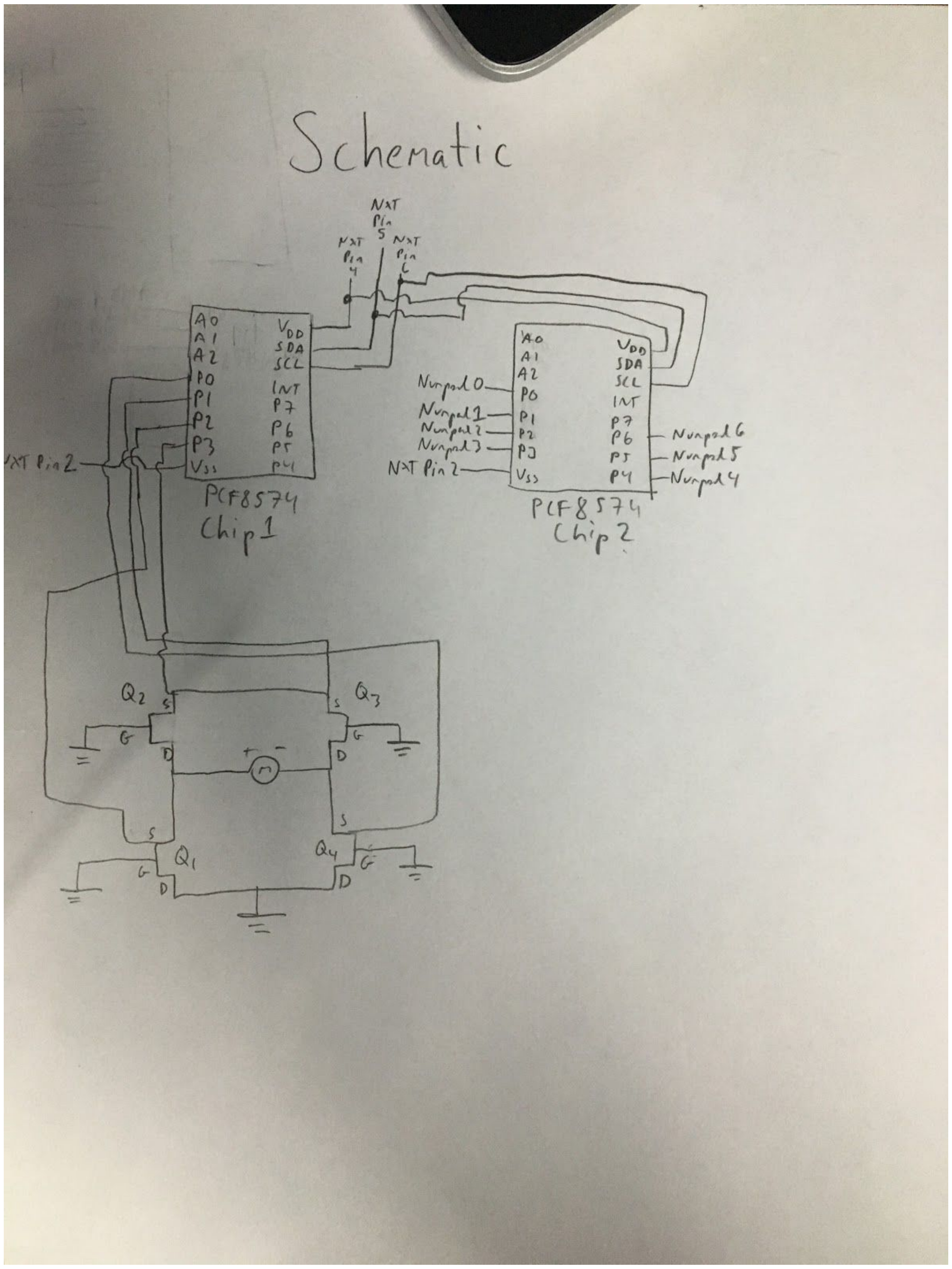
NXT
Adapter:https://www.generationrobots.com/en/401251-nxt-breadboard-adapter-dexter-industries.html

Wires:https://www.amazon.com/EDGELEC-Breadboard-Optional-Assorted-Multicolored/dp/B07GD2BWPY/ref=sxin_3_ac_d_rm?ac_md=1-1-anVtcGVyIHdpcmVz-ac_d_rm&crid=TPHWM5614WAI&keywords=breadboard%2Bwires&pd_rd_i=B07GD2BWPY&pd_rd_r=817a09c7-d444-4de0-ba4a-0a6dce2c4d56&pd_rd_w=an79B&pd_rd_wg=76Bnz&pf_rd_p=6d29ef56-fc35-411a-8a8e-7114f01518f7&pf_rd_r=7PCCSNKCF71J9N9TB7W0&qid=1575240281&sprefix=breadbo%2Caps%2C249&th=1

PART DESCRIPTION	VENDOR	PART	PRICE (2019) in $	QTY
IRF9530 MOSFET transistor	IOR	PD-9.320Q	1.81	2
IRF510 MOSFET transistor	IOR	PD-9.325Q	1.46	2
12-Key Keypad	Sparkfun	COM-14662	4.50	1
9V Battery	Duracell	MN16RT4Z	6.21	1
Motor	Pololu	1117	1.79	1
PCF8574	NXP	PCF8574P,112	3.48	2
Wires	Amazon	B07GD2BWPY	~0.05	35
NXT Adapter	Dexter	A-000000-00583	18.31	1

Schematic

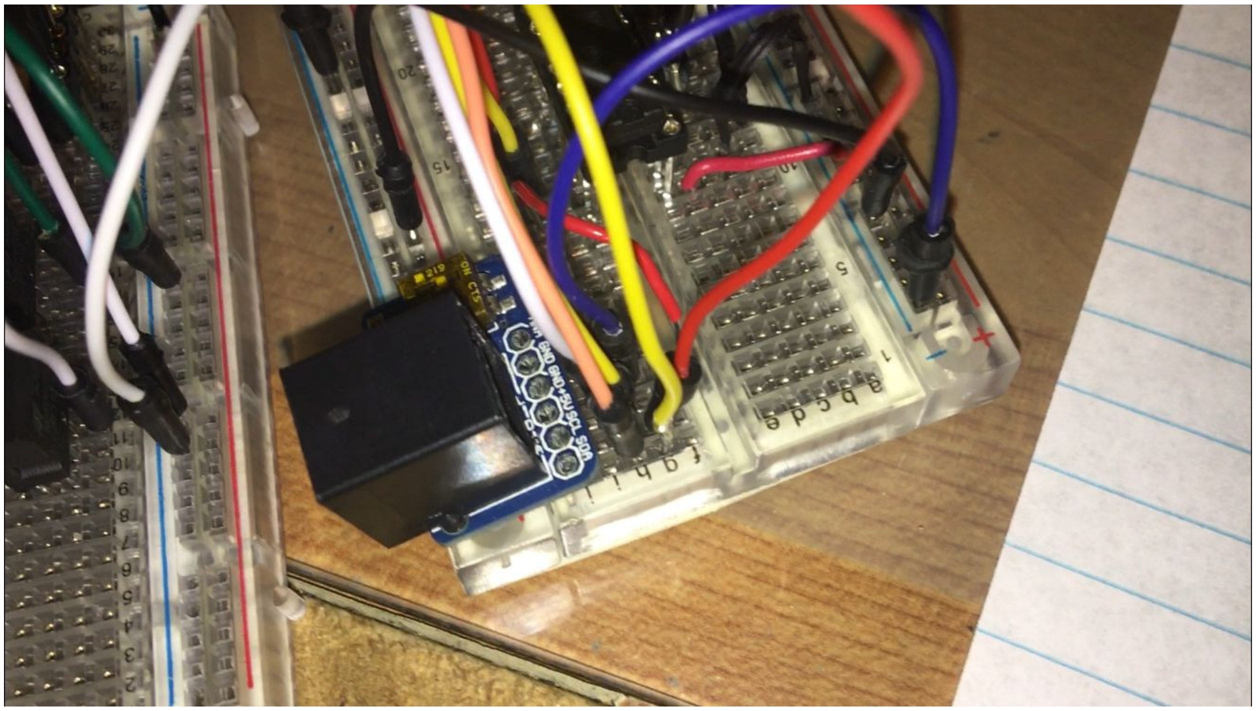
This section gives step-by-step instructions along with photos to construct a keypad-controlled motor. A schematic to construct this keypad-controlled motor is shown here



(Add hyperlink to PDF of schematic) is the Acrobat file of the same schematic.

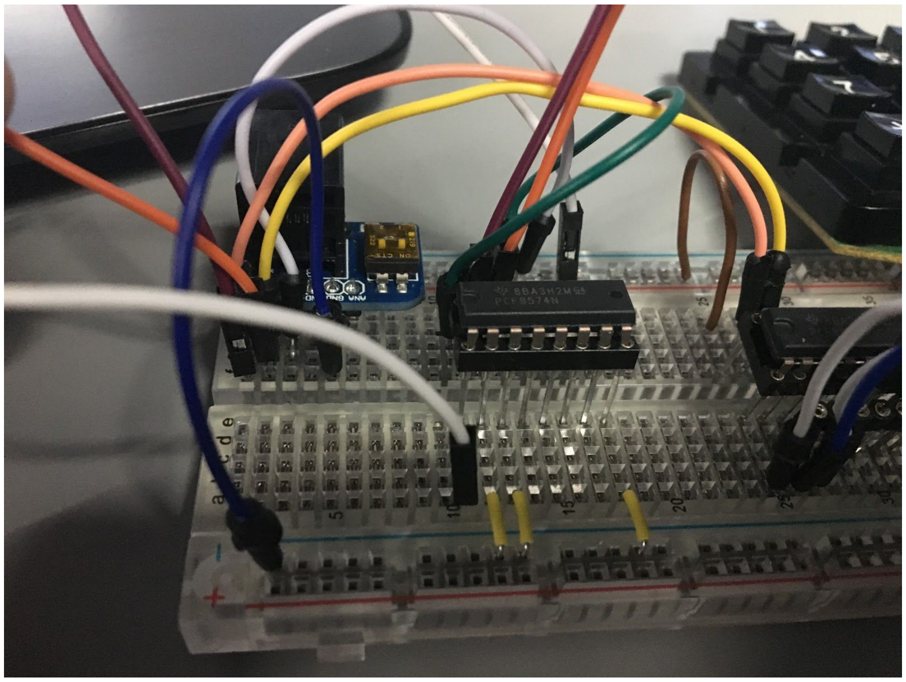
Step 1

Attach NXT Adapter to PCF8574 chips accordingly to a large breadboard. Leave digital lines D0-D6 empty for use later on both chips. Lines D7 and INT will not be used.



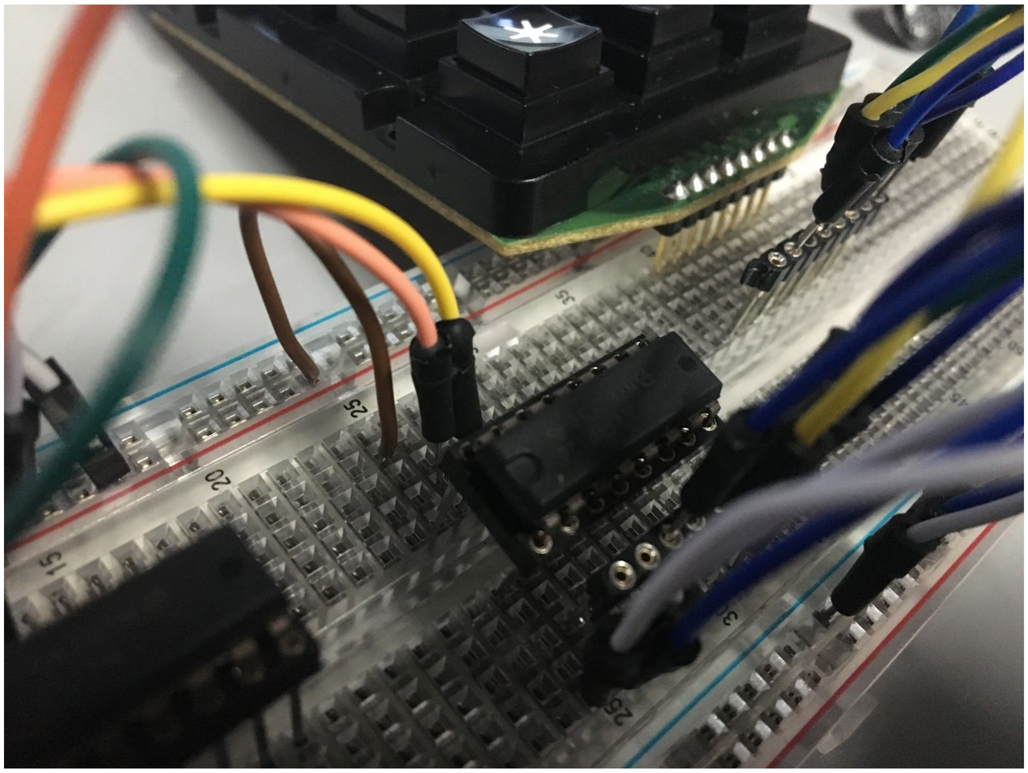
Step 2:

For the first PCF8547 chip, digital lines D0-D3 will be occupied by the G lines from the H-Bridge that will be constructed later. The A0 pin is connected to the 5V pin from the NXT adapter as well.



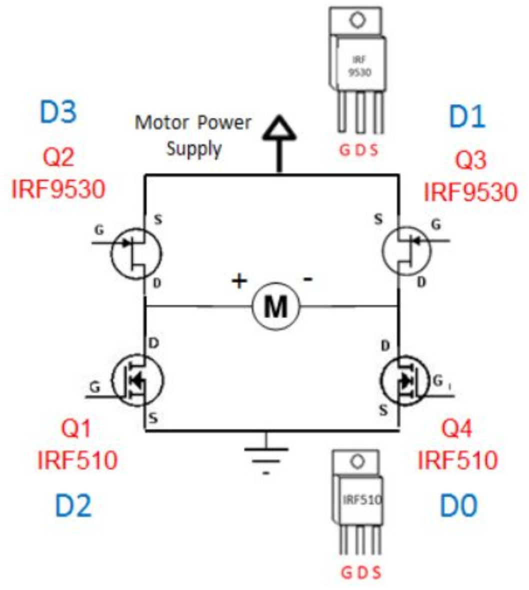
Step 3

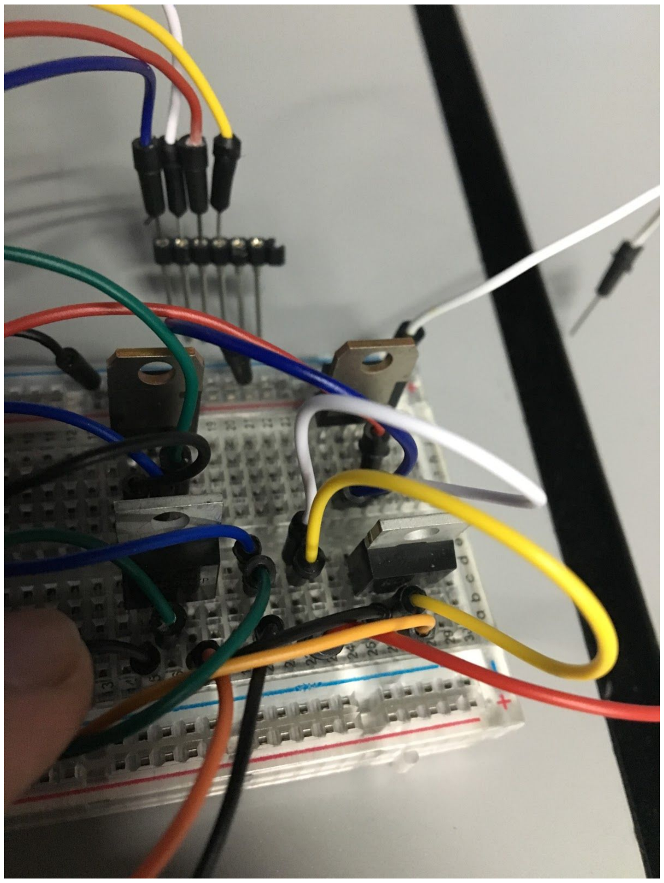
The second PCF8574 is set up similarly to the previous chip, except the A0 chip is connected to ground. Pins D0-D6 will be used to accept inputs from the 12-key keypad. D0-D3 is connected to Y1-Y4 and D4-D6 is connected to X1-X3.

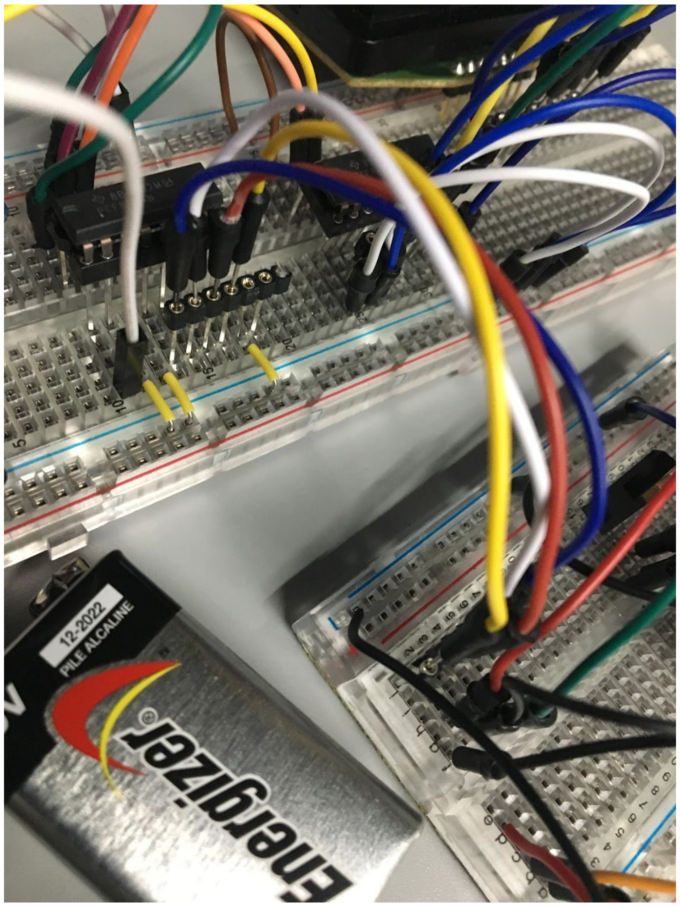


Step 4:

On a smaller breadboard, construct an H-Bridge using 2 IRF510 and 2 IRF9530 MOSFET transistors. A general schematic of how the H-Bridge is set up is provided below. The ground used in this H-Bridge should be connected to the ground from the larger breadboard to make a common ground. The G lines from the transistors should be connected to the first PCF8574 chip with the appropriately labeled pin.

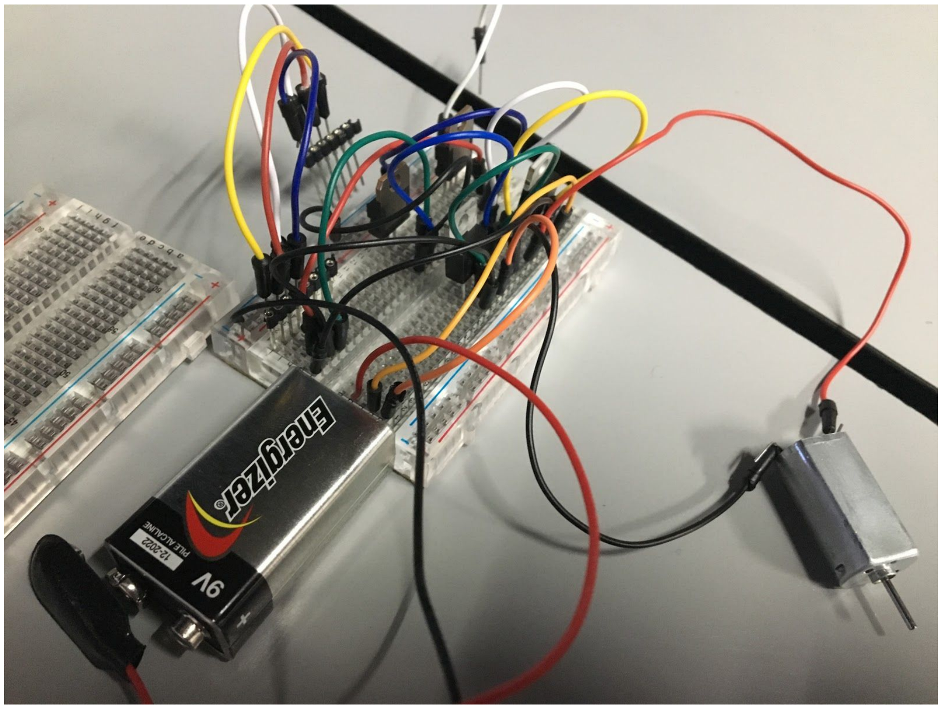





12-2022
PILE ALCALINE
Energizer

Step 5:

To get the H-Bridge to supply enough power to the motor, a 9V battery must be connected to the S lines on the IRF9530 transistors. The motor is connected in parallel to the D lines from the 4 transistors.



Here is a video of the H-bridge in operation: https://youtu.be/Q1hHf3rv4dw

Programming

The source code to (fill in the blank) is provided below:

To be compiled with (BricX Command Center)

```
// FILE: HBridge_Numpad_1b.nxc

// AUTH: Logan T. Arias

// DATE: 12/02/19

// VERS: 1.0

// DESC: PCF8574A receives an input from a numpad, and outputs results based upon it to an
H-Bridge

// NOTE: Uses PCF8574A chip (hence address A2-A1-A0 set to 0-0-0 hence 0x70



// Pressing 2, 4, 6, 8 on the numpad will cause a motor to rotate forward, backward,

// brake, or freely rotate. All other values will terminate the program.



#define I2Cport S1 // Port number

#define I2CAddr8574 x040 // I2C address x040 8574 or 0x70 for 8574A

#define I2CAddr8574a x042 // I2C address x042 8574 or 0x72 for 8574A


// Global variables

// array variables (since NXC's I2C functions take array variables
byte WriteBuf[] = {I2CAddr8574, 0x00}; // sets up the first PCF8574A for writing

byte WriteBuf2[] = {I2CAddr8574a, 0x00}; // sets up the second PCF8574A for writing

byte ReadBuf[]; // data received from the first PCF8574A.

byte ReadBuf2[]; // data received from the second PCF8574A.

int RdCnt = 1; // number of bytes to read for both chips


// This code enables the PCF8574A to receive inputs from a numpad.


long GetKey()

{

  WriteBuf[1] = 0xEF; // Col 1: 0xEF = 239 decimal = 1110 1111.  Sets Col 1 HI

  I2CBytes(I2Cport, WriteBuf, RdCnt, ReadBuf);

  if(ReadBuf[0]==0xEE) return(1); // Row 1; "1" key. 0xEE = 238 decimal = 1110 1110. If HI,
then "1" key

  if(ReadBuf[0]==0xED) return(4); // Row 2: "4" key

  if(ReadBuf[0]==0xEB) return(7); // Row 3; "7" key

  if(ReadBuf[0]==0xE7) return(14); // Row 4; "*" key


    WriteBuf[1] = 0xDF; // Col 2: 0xDF = 223 decimal = 1101 1111.  Sets Col 2 HI

  I2CBytes(I2Cport, WriteBuf, RdCnt, ReadBuf);

  if(ReadBuf[0]==0xDE) return(2); // Row 1; "2" key. 0xEE = 238 decimal = 1101 1110. If HI,
then "2" key
```

```
  if(ReadBuf[0]==0xDD) return(5); // Row 2: "5" key

  if(ReadBuf[0]==0xDB) return(8); // Row 3; "8" key

  if(ReadBuf[0]==0xD7) return(0); // Row 4; "0" key


  WriteBuf[1] = 0xBF; // Col 3: 0xBF = 191 decimal = 1011 1111.  Sets Col 3 HI

  I2CBytes(I2Cport, WriteBuf, RdCnt, ReadBuf);

  if(ReadBuf[0]==0xBE) return(3); // Row 1; "3" key. 0xEE = 238 decimal = 1011 1110. If HI,
then "3" key

  if(ReadBuf[0]==0xBD) return(6); // Row 2: "6" key

  if(ReadBuf[0]==0xBB) return(9); // Row 3; "9" key

  if(ReadBuf[0]==0xB7) return(15); // Row 4; "#" key


  return(-1); // return -1 if no key pressed

} // end of GetKey




task main()

{


 long key;

 long value = 0;
```

```
SetSensorLowspeed (I2Cport); // PCF8574A connect to NXT on S1

int decimalNumber = 0; // Value written by PCF8574A


while(true) { // endless loop
  while(GetKey() < 0); // do nothing
  key = GetKey(); // get key value
  switch(key) { // case based on key value

   case -1: // no key, do nothing
    break;
   case 2: // "2" key
    decimalNumber = 10; // 10 = 1010 = FREE ROTATION
    WriteBuf2[0] = I2CAddr8574a;
    WriteBuf2[1] = decimalNumber;
    I2CBytes(S1, WriteBuf2, RdCnt, ReadBuf2);
    PlayTone(300,5); // make key pressed sound
   case 4: // "4" key
    decimalNumber = 12; // 12 = 1100 = FORWARD ROTATION
    WriteBuf2[0] = I2CAddr8574a;
    WriteBuf2[1] = decimalNumber;
    I2CBytes(S1, WriteBuf2, RdCnt, ReadBuf2);
    PlayTone(1100,5); // make key pressed sound
    break;
   case 6: // "6" key
    decimalNumber = 3; // 3 = 0011 = BACKWARD ROTATION
    WriteBuf2[0] = I2CAddr8574a;
    WriteBuf2[1] = decimalNumber;
    I2CBytes(S1, WriteBuf2, RdCnt, ReadBuf2);
    PlayTone(2200,5); // make key pressed sound
    break;
   case 8: // "8" key
    decimalNumber = 15; // 15 = 1111 = BRAKE
    WriteBuf2[0] = I2CAddr8574a;
    WriteBuf2[1] = decimalNumber;
    I2CBytes(S1, WriteBuf2, RdCnt, ReadBuf2);
    PlayTone(3600,5); // make key pressed sound
    break;
   case 15: // #: quit
    Stop(true);
    break;
   //default:
  // Stop(true);
```

```
    //break;

    } // end switch


    while(GetKey() >= 0); // wait for key to be unpressed


  } // end of while

} // end main
```

---

fileName.C Fuller Code Description

The HBridge_Numpad_1b.nxc operates as follows. The code sets up two PCF8574 chips to read and write. The first chip reads an input from a programmed numpad. The values on the numpad are determined from the parts of the pin where a short occurs; this could be converted into decimal and read from the chip. The code was set up so that the combination of pins for each button on the numpad could be determined.

The second chip takes the data that was read from the first chip, and uses that to determine the kind of signal that should be output to the four transistors. Depending on which combination of transistors was active, the motor will rotate one way or the other. The input from the numpad was determined from the first chip, which could be separated into different cases. This was used to determine what kind of output should occur, which would in turn affect how the DC motor would behave.

Final Words

This tutorial's objective was to construct an H-bridge and control it using a keypad and a PCF8574 chip. Building a circuit, programming the PCF8574 chip, and using digital inputs for controlling the behavior of a motor was accomplished with the included steps and documentation. Once the concepts were conveyed, the reader could reliably construct their own H-Bridge and program it to work. This will provide them with valuable skills in the future related to controls, automation, and circuitry.

Speculating future work, derived from this tutorial, includes controlling more complicated motor systems using H-bridges. In the big picture, the problem of machine control and modulation can be solved with this tutorial.

E-mail us at ariasm6@unlv.nevada.edu and ariasl1@unlv.nevada.edu