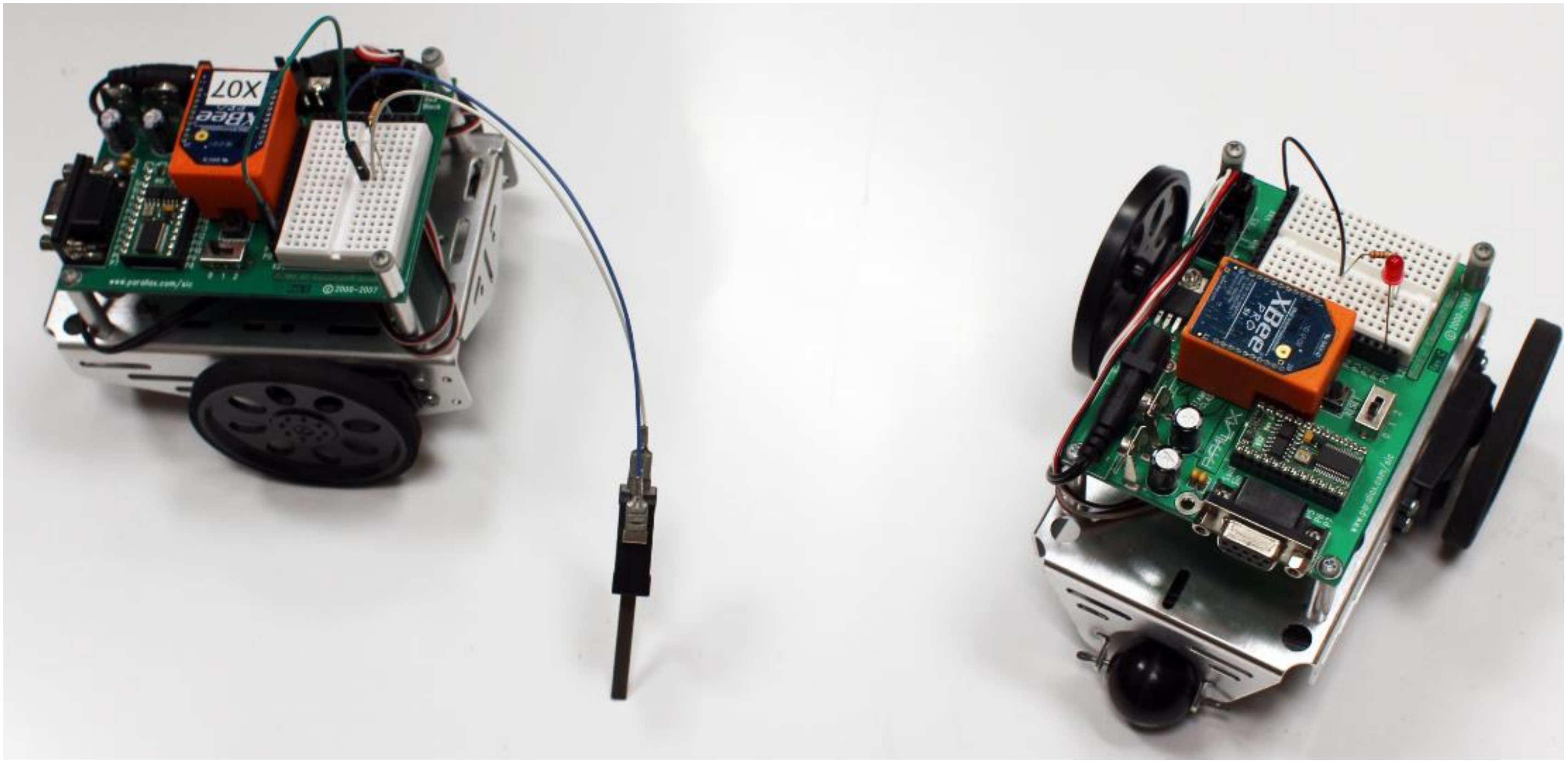
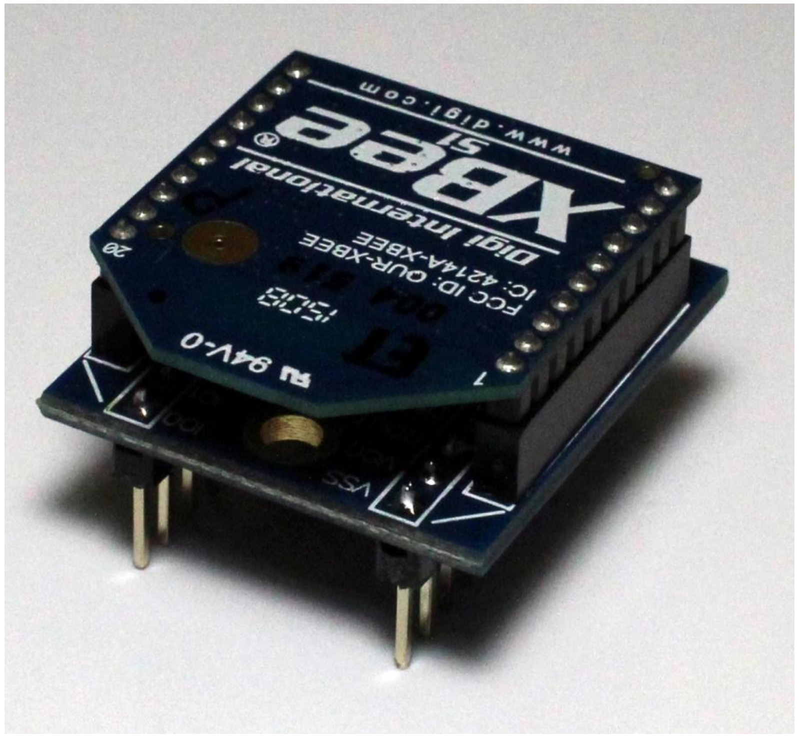
wireless control of an LED

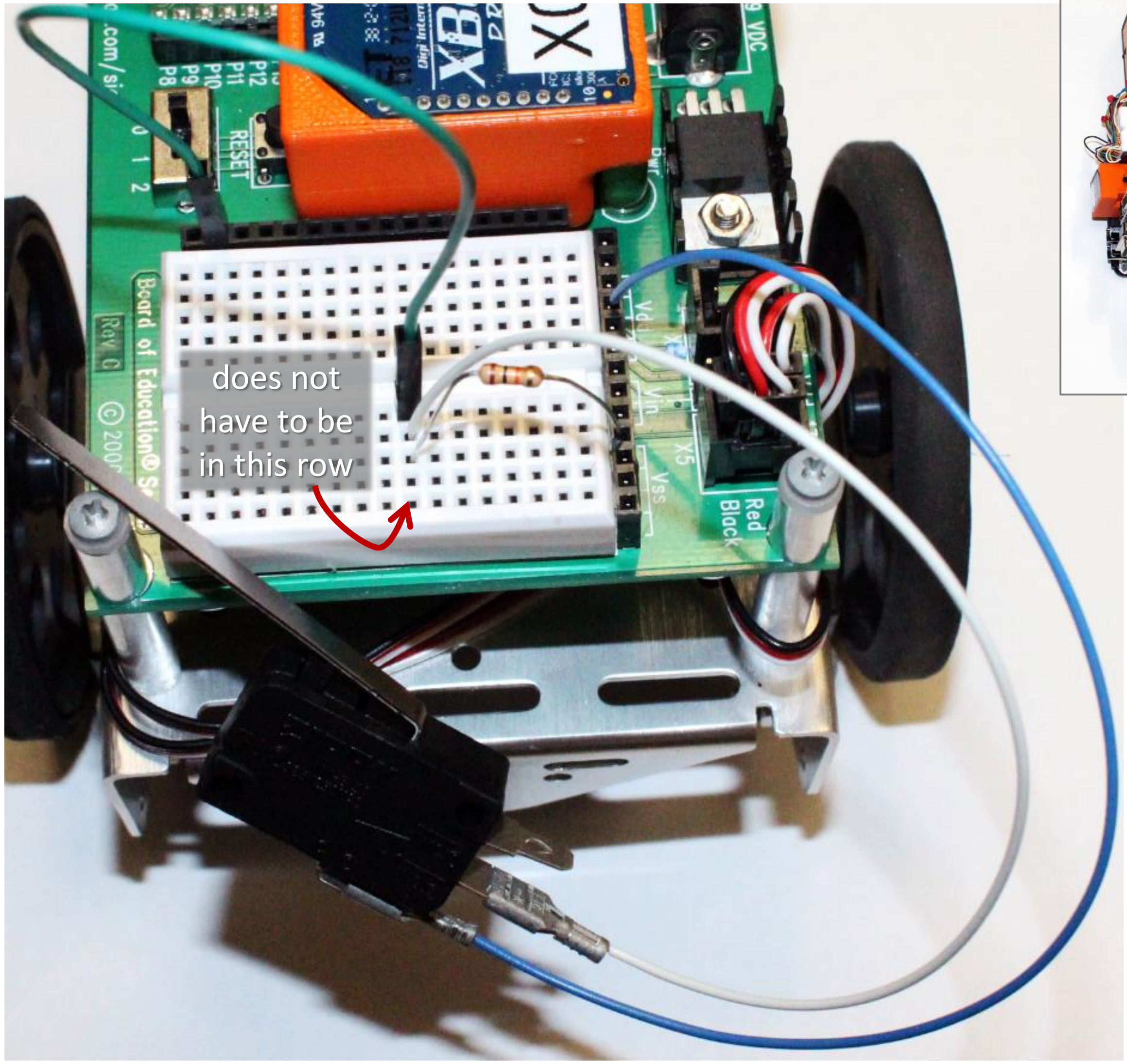


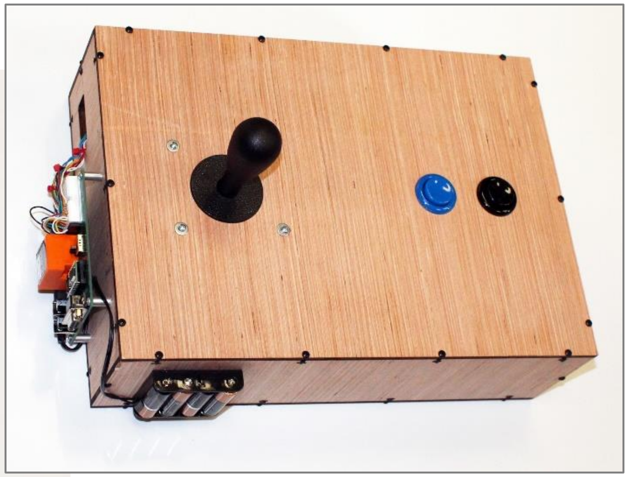
the XBee transceiver

- transmitter: sends radio waves
- receiver: receives radio waves
- transceiver: sends AND receives



hooking up your switch – transmitter side





use a switch from your control box
(forward or left)

program the transmitter

```
' {$STAMP BS2}
' {$PBASIC 2.5}

X          VAR Byte

CHANNEL    CON $0C     'channel
BAUD       CON 84      ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX         PIN 0       ' Receive Pin
TX         PIN 2       ' Transmit Pin

GOSUB EstablishConnection

' Read in a value
DO
  X = 0
  IF IN3=1 THEN X=1


  DEBUG "X=", DEC X, CR             ' Print out "x="

    SEROUT  TX, BAUD, [DEC X, CR, CR]  ' Send value of X

  ' Second CR is added byte buffer for flow control example
  PAUSE 20
LOOP
END

EstablishConnection:
  HIGH TX
  DEBUG CLS, "Configuring XBee..."
  PAUSE 2000                              ' Guard time for command sequence
  SEROUT TX, BAUD,["+++"]                 ' Enter command mode
  PAUSE 2000                              ' Guard time for command sequence
  SEROUT TX, BAUD, ["ATCH ", HEX CHANNEL, CR]  ' Set channel
  PAUSE 2000
  DEBUG "Configuration Complete!",CR
  RETURN
```

main loop

Declares a variable X into which "instructions" can be stored

Sets a transmission channel. This channel is specified with a Hexadecimal number. We will supply your team with a unique value

This line establishes other settings for the transceiver that we don't need to discuss

Sets the pins used for transmitting (TX) and receiving (RX) data. Only TX is used for transmitter

This line calls a subroutine that will cause the transceiver modules to connect with each other

Resets the value stored in X each time the code loops

*this IF statement only updates the value stored in X if pin 3 has 5V applied to it (i.e. **HIGH** or **1**)*

shows what value has been stored in X on the DEBUG terminal

SEROUT sends the value stored in X serially thru the wire plugged into P2 (TX) to the transmitter XBEE, which sends it wirelessly to the receiver XBEE

allows time for the transmission to take place before sending something else

As the name implies, this code causes the transceiver modules to connect with each other. It can/should be used without editing. The message indicating "Configuration Complete" may appear a few seconds before you will actually be able to drive your vehicle

program the receiver

```
' {$STAMP BS2}
' {$PBASIC 2.5}

RX          PIN 0          ' Receive Pin
TX          PIN 2          ' Transmit Pin
X           VAR Byte
counter     VAR Byte
CHANNEL     CON $0C
BAUD        CON 84

GOSUB EstablishConnection

DO
  SERIN RX, BAUD, [DEC X]    ' Receive Data

  DEBUG "X="
  DEBUG DEC X, CR

  IF X=1 THEN HIGH 1
  IF X=0 THEN LOW 1

LOOP
END

EstablishConnection:
  HIGH TX
  DEBUG CLS, "Configuring XBee..."
  PAUSE 2000
  SEROUT TX, BAUD, ["+++"]
  PAUSE 2000
  SEROUT TX, BAUD, ["ATCH ", HEX CHANNEL, CR]
  PAUSE 2000
  DEBUG "Configuration Complete!", CR
  RETURN
```

main loop

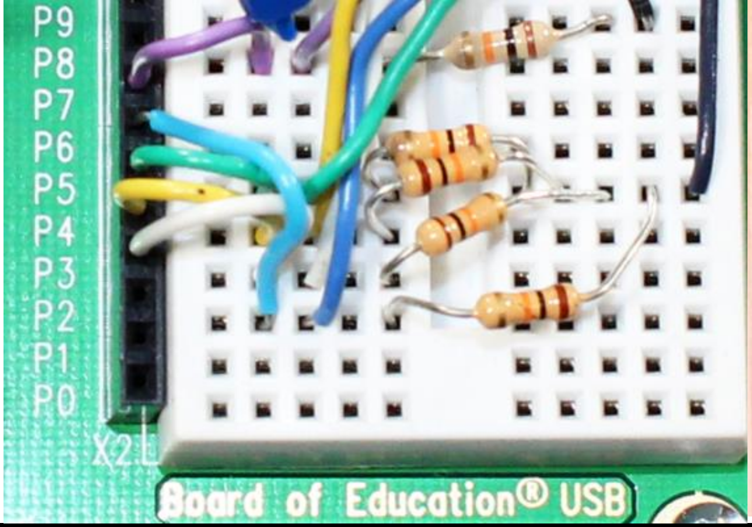
sets a transmission channel. Make sure this number matches the one programmed into the transmitter

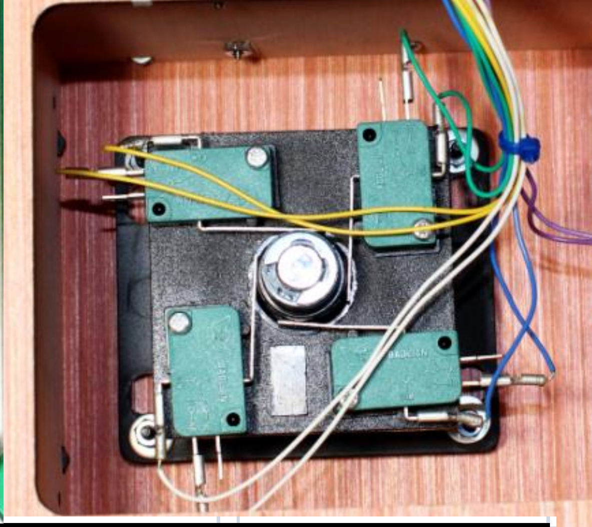
when the receiver XBEE receives a value wirelessly, it sends that value serially to pin P0. SERIN receives the value and stores it into X

if a computer is hooked up to the receiver, a DEBUG terminal can show the values received

use IF statements to turn an LED connected to pin P1 on or off based on the received value

dealing with multiple switch states





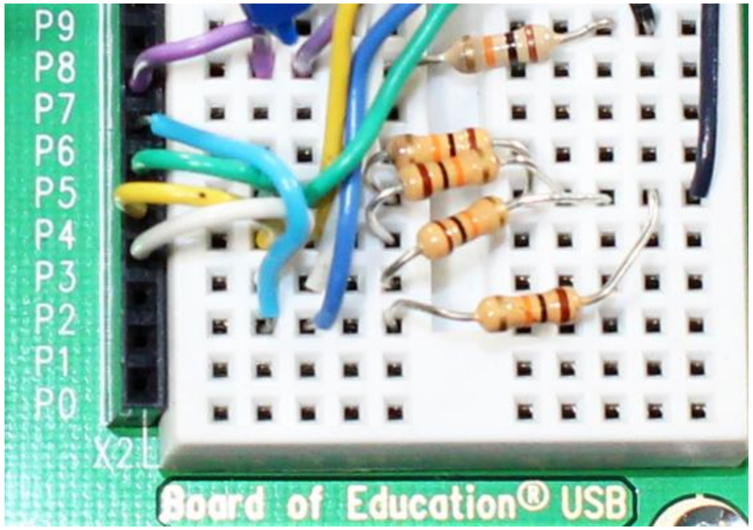
Joystick Switch:	Forward	Backward	Right	Left	Without Key Depressed:
Transmitter Variable:	F	B	R	L	
Wire Color:	blue	yellow	green	white	Character
Input Pin:	P6	P4	P3	P5	Transmitted
Straight Forward	1	0	0	0	a
Straight Backward	0	1	0	0	b
Right Turn (forward)	1	0	1	0	c
Left Turn (forward)	1	0	0	1	d
Right Turn (backward)	0	1	1	0	e
Left Turn (backward)	0	1	0	1	f
Steer Right (not moving)	0	0	1	0	g
Steer Left (not moving)	0	0	0	1	h
No Action	0	0	0	0	n

joystick programming

```
' {$STAMP BS2}
' {$PBASIC 2.5}

X          VAR Byte
F          VAR Bit
B          VAR Bit
L          VAR Bit
R          VAR Bit

CHANNEL    CON $0C
BAUD       CON 84

RX         PIN 0
TX         PIN 2

GOSUB EstablishConnection
```

```
DO
F=IN6
B=IN4
L=IN3
R=IN5

IF F=1 AND B=0 AND R=0 AND L=0 THEN
X="a"
ELSEIF F=0 AND B=1 AND R=0 AND L=0 THEN
X="b"
ELSEIF F=1 AND B=0 AND R=1 AND L=0 THEN
X="c"
ELSEIF F=1 AND B=0 AND R=0 AND L=1 THEN
X="d"
ELSEIF F=0 AND B=1 AND R=1 AND L=0 THEN
X="e"
ELSEIF F=0 AND B=1 AND R=0 AND L=1 THEN
X="f"
ELSEIF F=0 AND B=0 AND R=1 AND L=0 THEN
X="g"
ELSEIF F=0 AND B=0 AND R=0 AND L=1 THEN
X="h"
ELSE
X="n"
ENDIF

  DEBUG "X="
  DEBUG X, CR

  SEROUT  TX, BAUD, [DEC X, CR, CR]

  PAUSE 10
LOOP
END
```

what about the button?



Joystick Switch:	Forward	Backward	Right	Left	Without Key	With Key
Transmitter Variable:	F	B	R	L	Depressed:	Depressed:
Wire Color:	blue	yellow	green	white	Character	Character
Input Pin:	P6	P4	P3	P5	Transmitted	Transmitted
Straight Forward	1	0	0	0	a	A
Straight Backward	0	1	0	0	b	B
Right Turn (forward)	1	0	1	0	c	C
Left Turn (forward)	1	0	0	1	d	D
Right Turn (backward)	0	1	1	0	e	E
Left Turn (backward)	0	1	0	1	f	F
Steer Right (not moving)	0	0	1	0	g	G
Steer Left (not moving)	0	0	0	1	h	H
No Action	0	0	0	0	n	N

car programming

```
' {$STAMP BS2}
' {$PBASIC 2.5}

RX          PIN 0
TX          PIN 2
X           VAR Byte

center      CON 650
right       CON 750
left        CON 540
fullstop    CON 750
slowforward CON 680
slowback    CON 820

CHANNEL     CON $0C
BAUD        CON 84

GOSUB EstablishConnection
```

sample subroutine

```
SlowStraightForward:
PULSOUT 14, slowforward
PULSOUT 12, center
RETURN
```

```
DO
  SERIN RX, BAUD, [DEC X]

  DEBUG "X="
  DEBUG X, CR

IF X="a" THEN
GOSUB SlowStraightForward
ELSEIF X="b" THEN
GOSUB SlowStraightBackward
ELSEIF X="c" THEN
GOSUB SlowRightTurn
ELSEIF X="d" THEN
GOSUB SlowLeftTurn
ELSEIF X="e" THEN
GOSUB SlowRightBack
ELSEIF X="f" THEN
GOSUB SlowLeftBack
ELSEIF X="g" THEN
GOSUB SteerRight
ELSEIF X="h" THEN
GOSUB SteerLeft
ENDIF

LOOP
END
```