



DragonBoard™ 410c: Running a BreakerBall game

Lawrence King
Sr. Staff Engineer
Qualcomm Canada, Inc.

BreakerBall - a simple demo game

In this tutorial we will build a simple Game

There are two implementations of the game:

1. With the Linker Board and slide pot
2. With the 96Boards Sensors board, a rotary potentiometer and 2x16 LCD display

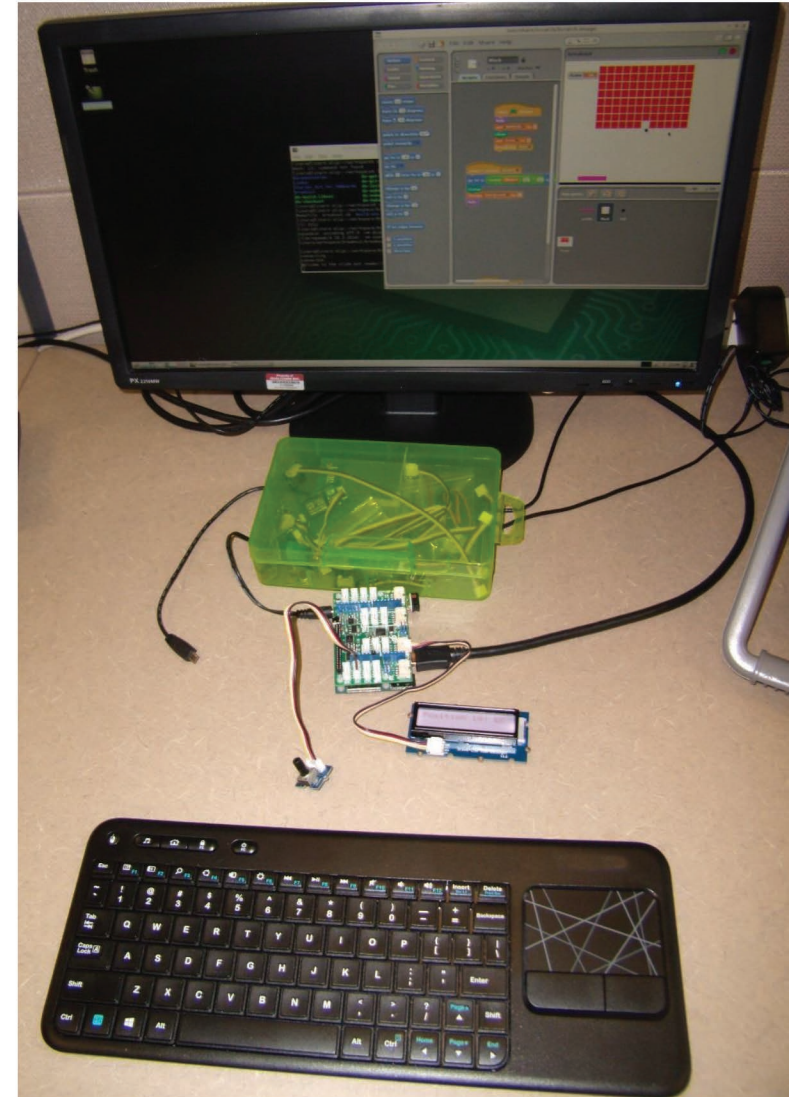
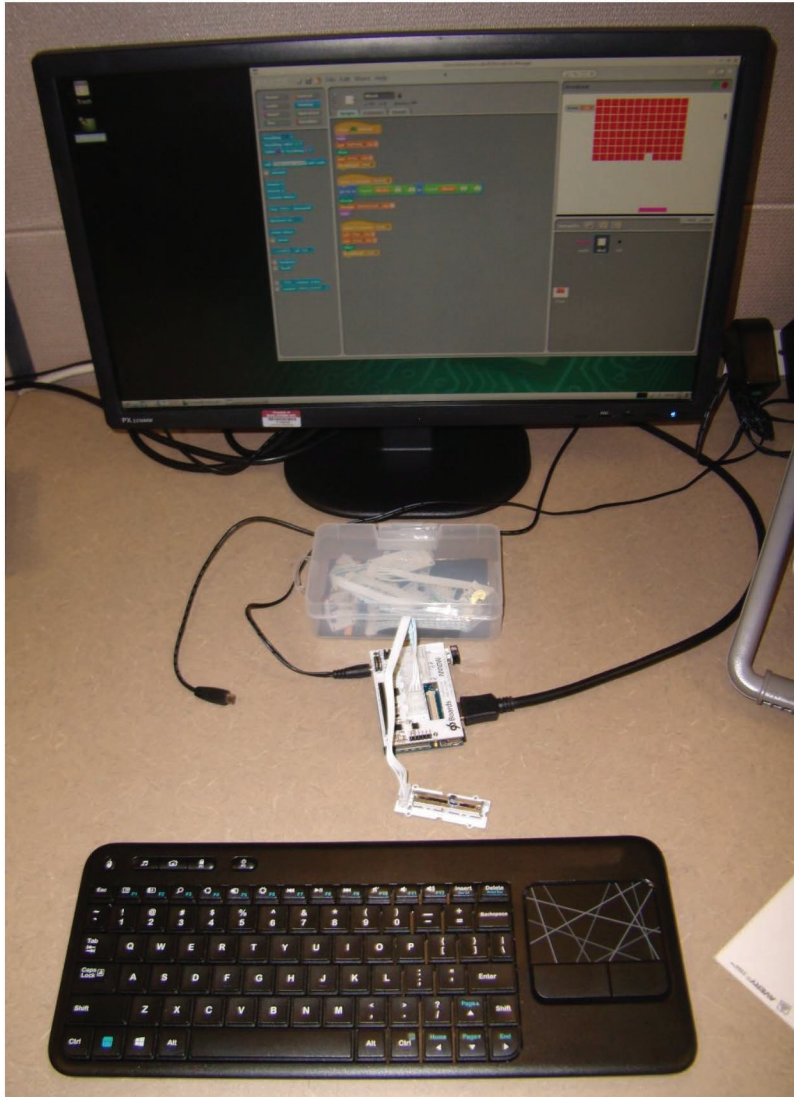


- All code was developed on the DragonBoard 410c using tools that are easily installed onto the board.

Only a few items are required for development:

- DragonBoard 410c, and 12V power supply.
- A mezzanine card kit (either Linker or 96Boards)
- A USB keyboard and USB mouse (I used a wireless combo keyboard/touchpad)
- A HDMI monitor (or the large screen TV in your living room)
- A Wi-Fi connection (to download the tools necessary)
- A 8GB or larger SD card (only needed if you are recompiling the kernel)

The Hardware setups



Setup the SW tools on the 410c

In a command window run the following commands:

Setup commands

```
sudo apt-get update
sudo apt-get dist-upgrade -u -y
sudo apt-get install -y man-db manpages manpages-dev
sudo apt-get install -y libpython-dev libpython3-dev python-dev
sudo apt-get install -y arduino-mk arduino
sudo apt-get install -y libmraa-dev libupm-dev libsoc-dev
sudo apt-get install -y scratch python-pip
pip install spidev
sudo apt-get clean
```

(I think this is all of the packages I used)

The Game

The Game is written in Scratch

The Game has 4 Scratch Elements

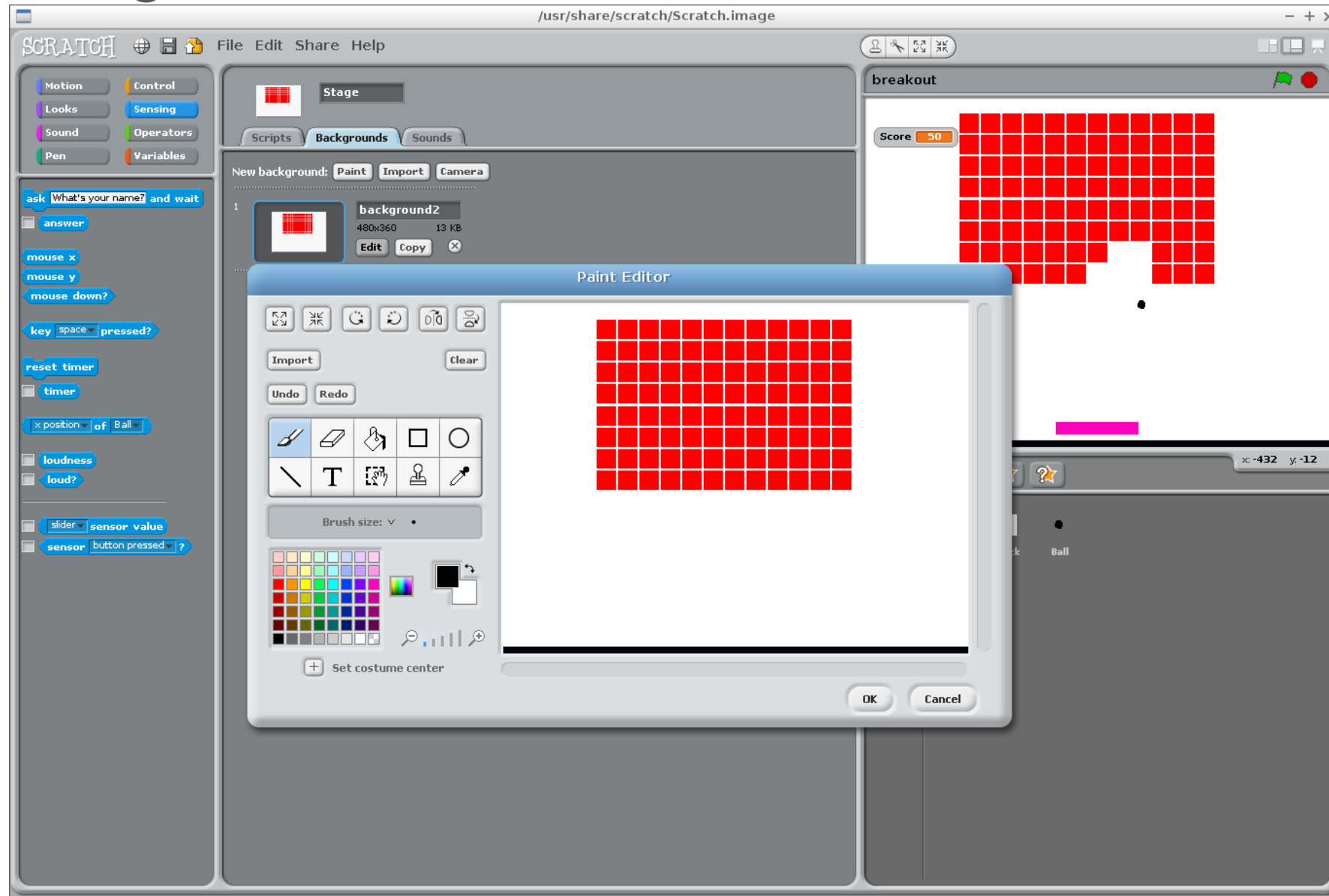
1. The Background which has the blocks drawn on it. This is just a simple image.
2. The paddle which moves back and forth depending on the position of the potentiometer
3. The Ball which bounces around
4. A white Block which is stamped onto the background effectively erasing the blocks

The Game is exactly the same for both implementations

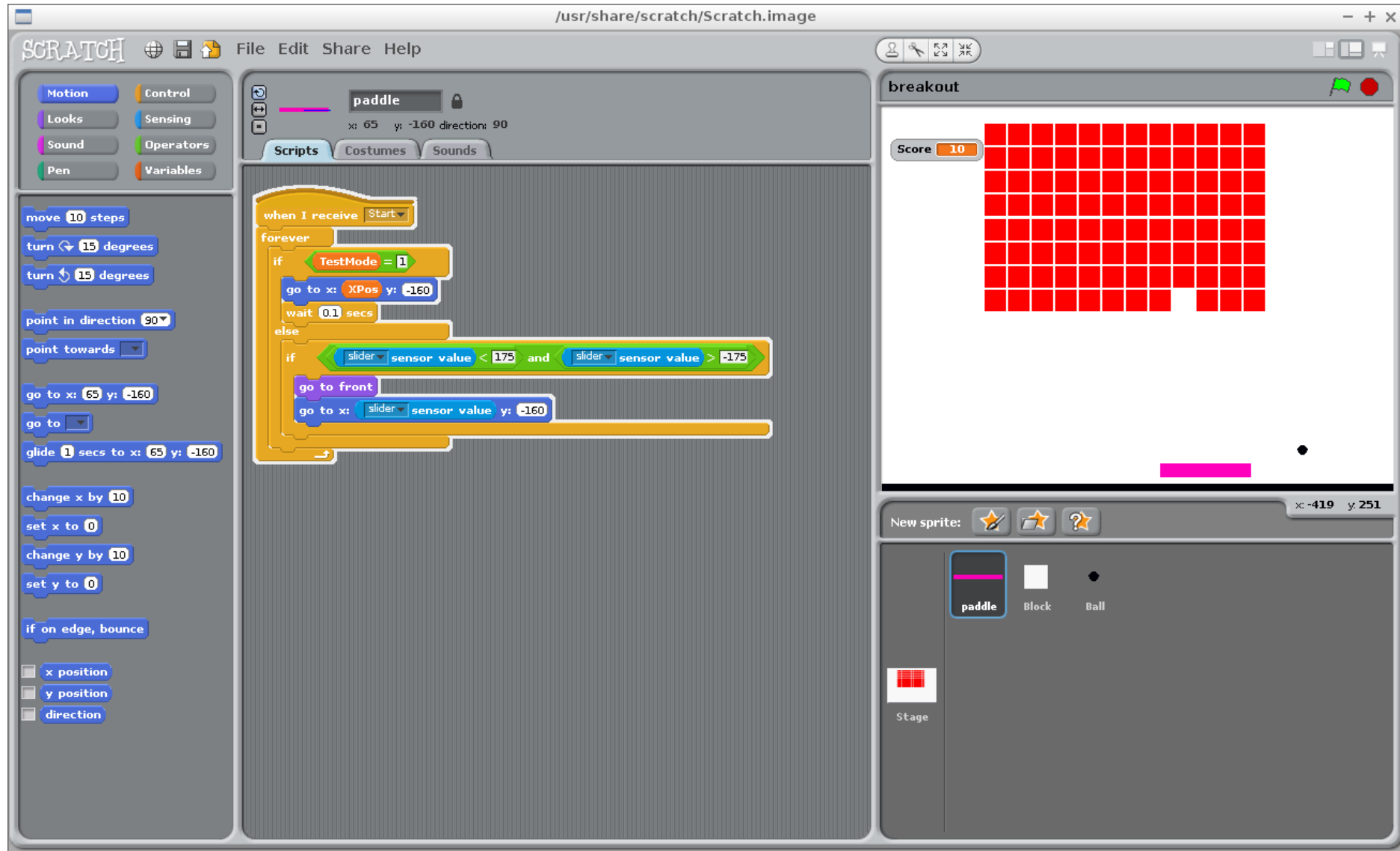
- Communications to and from Scratch is done via a local network port.
 - In Scratch, turn on remote sensor connections by right clicking the sensor variable and selecting “enable remote sensor connections”

Note: *not all Scratch implementations have this feature, however the version installed on the DragonBoard 410c does.*

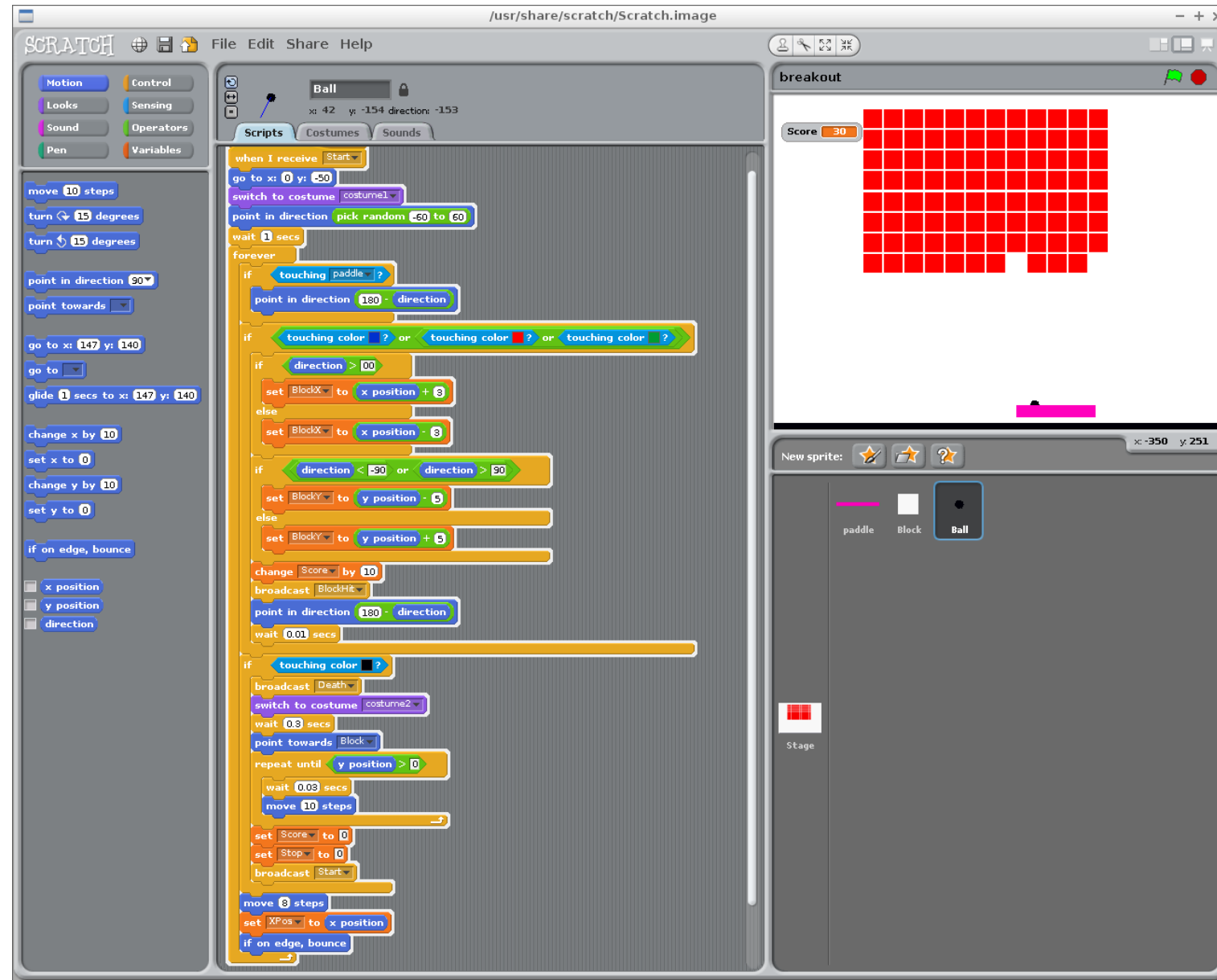
1. The Background



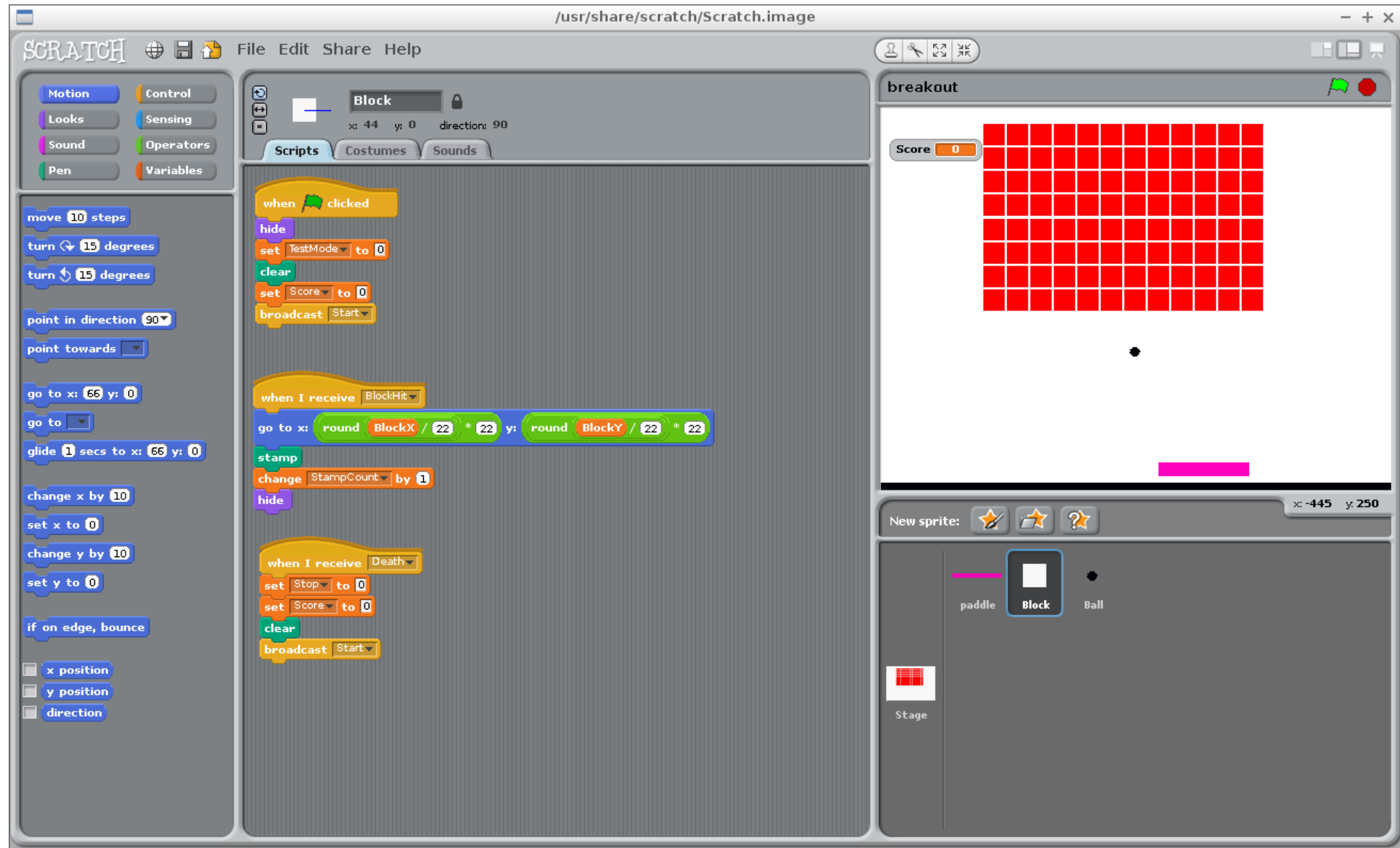
2. The Paddle



3. The Ball



4. The Block Stamper



The Hardware

Two versions of Hardware based in different Mezzanine Boards

The Linker Mezzanine Board version

- The Linker kit is available from Arrow.com for \$25
- Kit contains a Mezzanine board and several IO devices that just plug in with the included cables
- Major add-on chip on the mezzanine board is a 10-bit ADC
- A slide pot is one of the included peripherals in the kit.

The 96Boards Sensors Mezzanine board version

- The Sensors Mezzanine Board is available from Seeed for \$69
- Kit includes a mezzanine board and several IO devices that just plug in with the included cables
- Major add-on chip is an Arduino Uno on the board.
- A rotary pot and a 2x16 LCD display are included peripherals in the kit.

Note: *the IO devices in each kit are not easily interchangeable because they use different connectors.*

The SPI - Slide Pot reader

Python Program between spidev and Scratch port

The SPI slide pot reader is written in Python

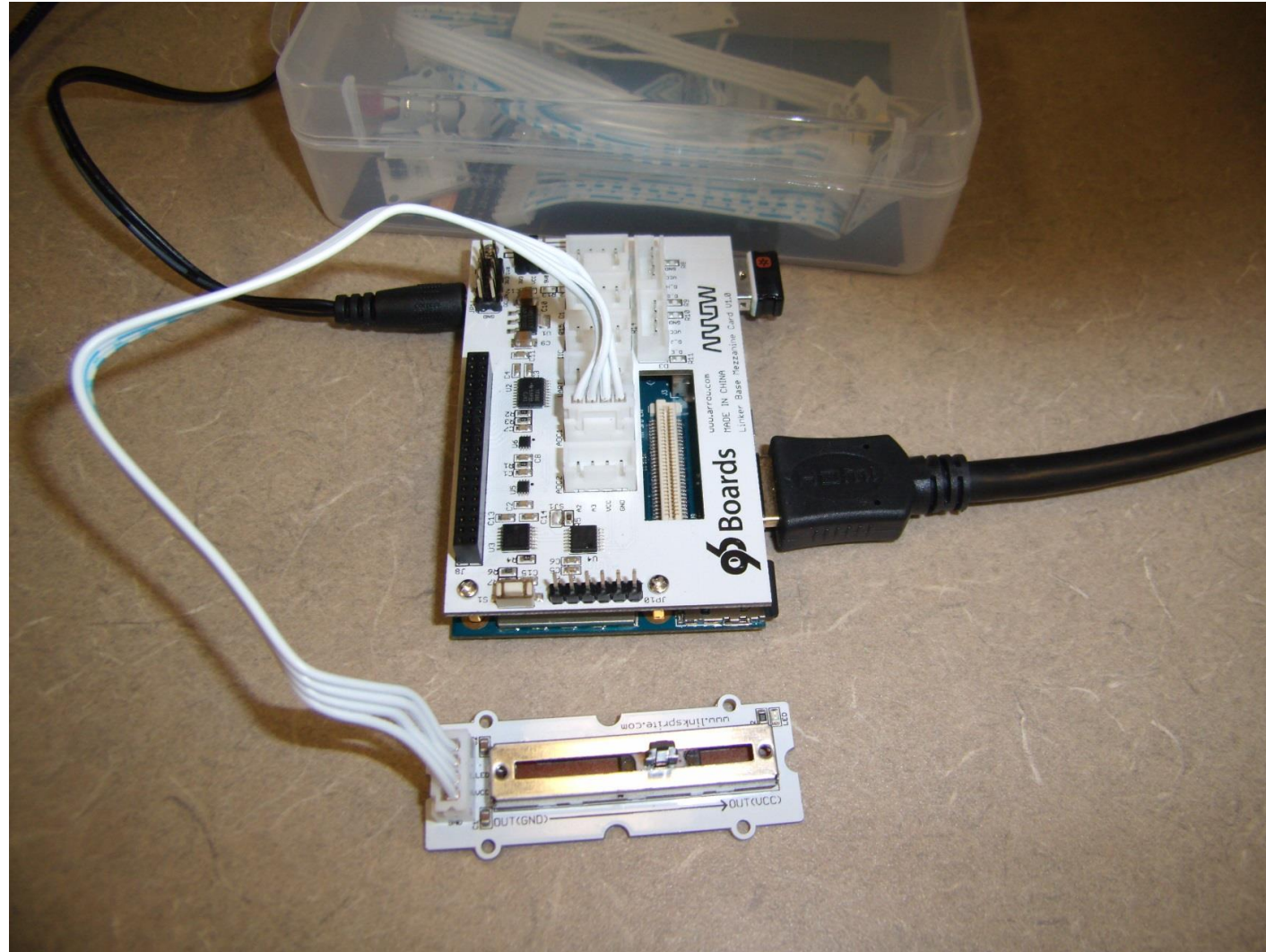
- Reads the slide pot position from /dev/spidev0.0
- Writes the slide pot position to the Scratch network port
- Repeats 20 times per second

Note: *The current release of Debian does not have spidev turned on by default (later versions should have it turned on). It is necessary to make a couple very minor changes and recompile the kernel.*

- Instructions to make the changes to enable SPI and then recompile the kernel are online at 96Boards.org
- ‘technically’ only the device tree is changed, but it is easier to update the entire kernel+devicetree image than to only update the device tree blob.

The Linker Mezzanine card

Set up the linker mezzanine card and the slide pot as shown



Python SPI reader program

```
linaro@linaro-alip: ~/workspace/breakout
File Edit Tabs Help
// Copyright (c) 2016 Lawrence King
//
// All rights reserved.
import serial
from array import array
import socket
import time
import sys
import spidev

PORT = 42001
HOST = '127.0.0.1'
if not HOST:
    sys.exit()

print("connecting...")
scratchSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
scratchSock.connect((HOST,PORT))
print("connected.")

spi=spidev.SpiDev()
spi.open(0,0)
spi.max_speed_hz=100000
channel_select=[0x01, 0x80, 0x00]

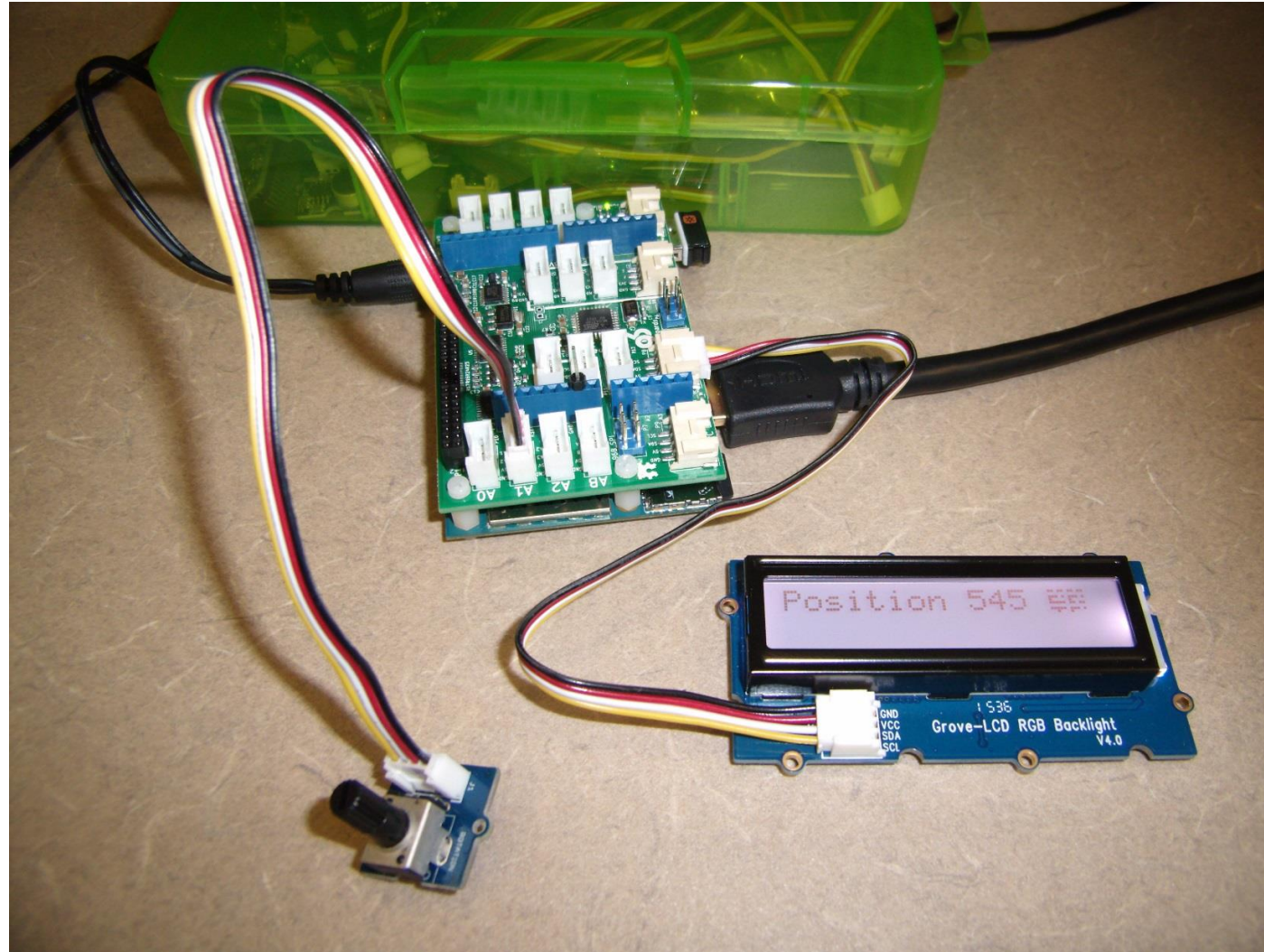
def sendScratchCommand(cmd):
    n = len(cmd)
    a = array('c')
    a.append(chr((n>>24) & 0xFF))
    a.append(chr((n>>16) & 0xFF))
    a.append(chr((n>>8) & 0xFF))
    a.append(chr((n) & 0xFF))
    scratchSock.send(a.tostring() + cmd)

if __name__ == '__main__':
    print("Welcome to the slide pot reader!!!")
    try:
        while True:
            adc_data = spi.xfer(channel_select)
            adc = ((adc_data[1]<<8)&0x300)|(adc_data[2]&0xFF)
            msg = str(((adc*351)/1024)-175)
            sendScratchCommand("sensor-update \"slider\" %s" % msg)
            print(msg)
            time.sleep(1.0/20)
    except KeyboardInterrupt:
        print("CTRL-C!! Exiting...")

~
(END)
```

The 96Boards Sensors Mezzanine card

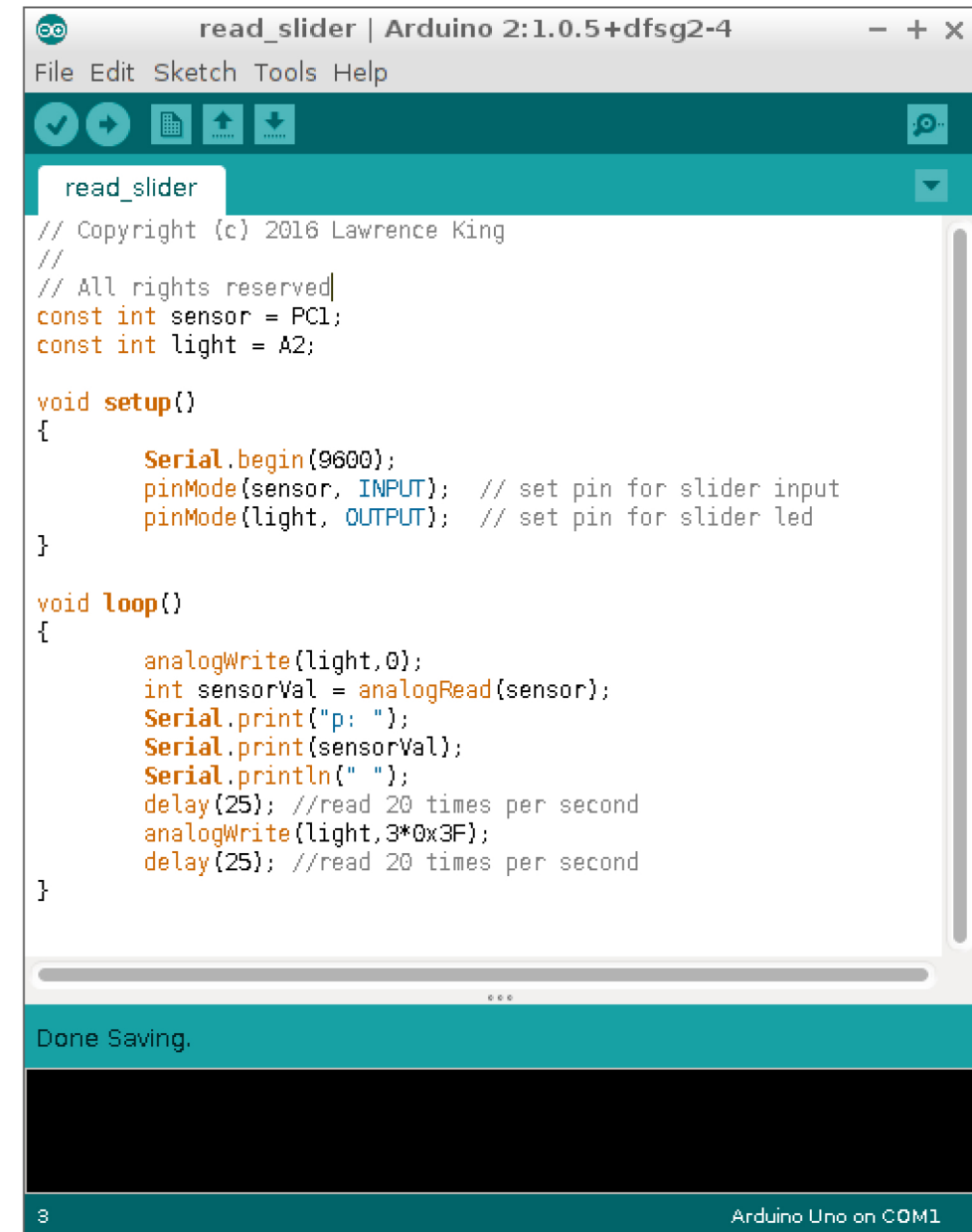
Set up the 96Boards mezzanine card, the display, and the rotary pot as shown:



The Arduino Program

Runs on the Arduino

- The Arduino sketch reads the position of the pot, formats it and sends it to the serial port.
- Prints the pot position to the serial port where it is received by the Python program.
- Repeats 20 times per second



The screenshot shows the Arduino IDE interface. The title bar reads "read_slider | Arduino 2:1.0.5+dfsg2-4". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for checking, running, uploading, and downloading. The main text area contains the following C++ code:

```
// Copyright (c) 2016 Lawrence King
//
// All rights reserved
const int sensor = PC1;
const int light = A2;

void setup()
{
    Serial.begin(9600);
    pinMode(sensor, INPUT); // set pin for slider input
    pinMode(light, OUTPUT); // set pin for slider led
}

void loop()
{
    analogWrite(light,0);
    int sensorVal = analogRead(sensor);
    Serial.print("p: ");
    Serial.print(sensorVal);
    Serial.println(" ");
    delay(25); //read 20 times per second
    analogWrite(light,3*0x3F);
    delay(25); //read 20 times per second
}
```

At the bottom of the IDE, there is a status bar that says "Done Saving." and a terminal window that is currently empty. The bottom right corner of the IDE shows "3" and "Arduino Uno on COM1".

The Arduino Slide Pot Reader Python program

Python Program - between Arduino serial port and Scratch

Receive the Arduino serial data and send to the Scratch port

- Waits for a line of text containing the potentiometer position
- Reformats the text
- Displays the potentiometer position on the LCD display
- Sends the potentiometer position to the Scratch port
- Repeats as fast as the Arduino is sending data

The Arduino Slide Pot Reader Python program

```
linaro@linaro-alip: ~/workspace/breakout
File Edit Tabs Help
# Copyright (c) 2016 Lawrence King
#
# All rights reserved
import serial
import pyupm_i2clcd
from array import array
import socket
import time
import sys

PORT = 42001
HOST = '127.0.0.1'
if not HOST:
    sys.exit()
print("connecting...")
scratchSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
scratchSock.connect((HOST,PORT))
print("connected.")

ard = serial.Serial('/dev/tty96B0', 9600)
lcd = pyupm_i2clcd.Jhd1313m1(0, 0x3e, 0x62)

def sendScratchCommand(cmd):
    n = len(cmd)
    a = array('c')
    a.append(chr((n>>24) & 0xFF))
    a.append(chr((n>>16) & 0xFF))
    a.append(chr((n>>8) & 0xFF))
    a.append(chr((n) & 0xFF))
    scratchSock.send(a.tostring() + cmd)

def showSlider(pos):
    lcd.clear()
    lcd.setCursor(0, 0)
    lcd.write("Position: " + pos)
    lcd.setColor(255, 180, 180)

if __name__ == '__main__':
    print("Welcome to the slide pot reader!!!")
    try:
        while True:
            ardOut = ard.readline()
            if ardOut.find("p:") != -1:
                ardSlider = ardOut.split('p:')[1]
                msg = str(((int(ardSlider)*351/1024)-175))
                showSlider(msg)
                sendScratchCommand("sensor-update \"slider\" %s" % msg)
            except KeyboardInterrupt:
                lcd.setColor(0,0,0)
                lcd.clear()
                print("CTRL-C!! Exiting...")
    slider ard.py (END)
```

Thank you



Follow us on:    

For more information, visit us at:

www.qualcomm.com & www.qualcomm.com/blog

Nothing in these materials is an offer to sell any of the components or devices referenced herein.

©2016 Qualcomm Technologies, Inc. and/or its affiliated companies. All Rights Reserved.

Qualcomm and DragonBoard are trademarks of Qualcomm Incorporated, registered in the United States and other countries. Other products and brand names may be trademarks or registered trademarks of their respective owners.

References in this presentation to “Qualcomm” may mean Qualcomm Incorporated, Qualcomm Technologies, Inc., and/or other subsidiaries or business units within the Qualcomm corporate structure, as applicable. Qualcomm Incorporated includes Qualcomm’s licensing business, QTL, and the vast majority of its patent portfolio. Qualcomm Technologies, Inc., a wholly-owned subsidiary of Qualcomm Incorporated, operates, along with its subsidiaries, substantially all of Qualcomm’s engineering, research and development functions, and substantially all of its product and services businesses, including its semiconductor business, QCT.