Smartphone Universal Remote, 2/14/2013

"Team Smart Universal: Colton Wells, Tin Nguyen"

Problem Definition

- Use a smarphone as a universal remote control.

- 2/11 Design
- 2/18 Order Parts
- 3/18 prototype remote

- $42 in parts
- 75 Engineering student hours

Solution Specifications

- A device will receive bluetooth signals from your smartphone and send an infrared signal to your TV

- Arduino Uno
- Blue tooth modem
- IR LED
- Android phone

Competitive Analysis

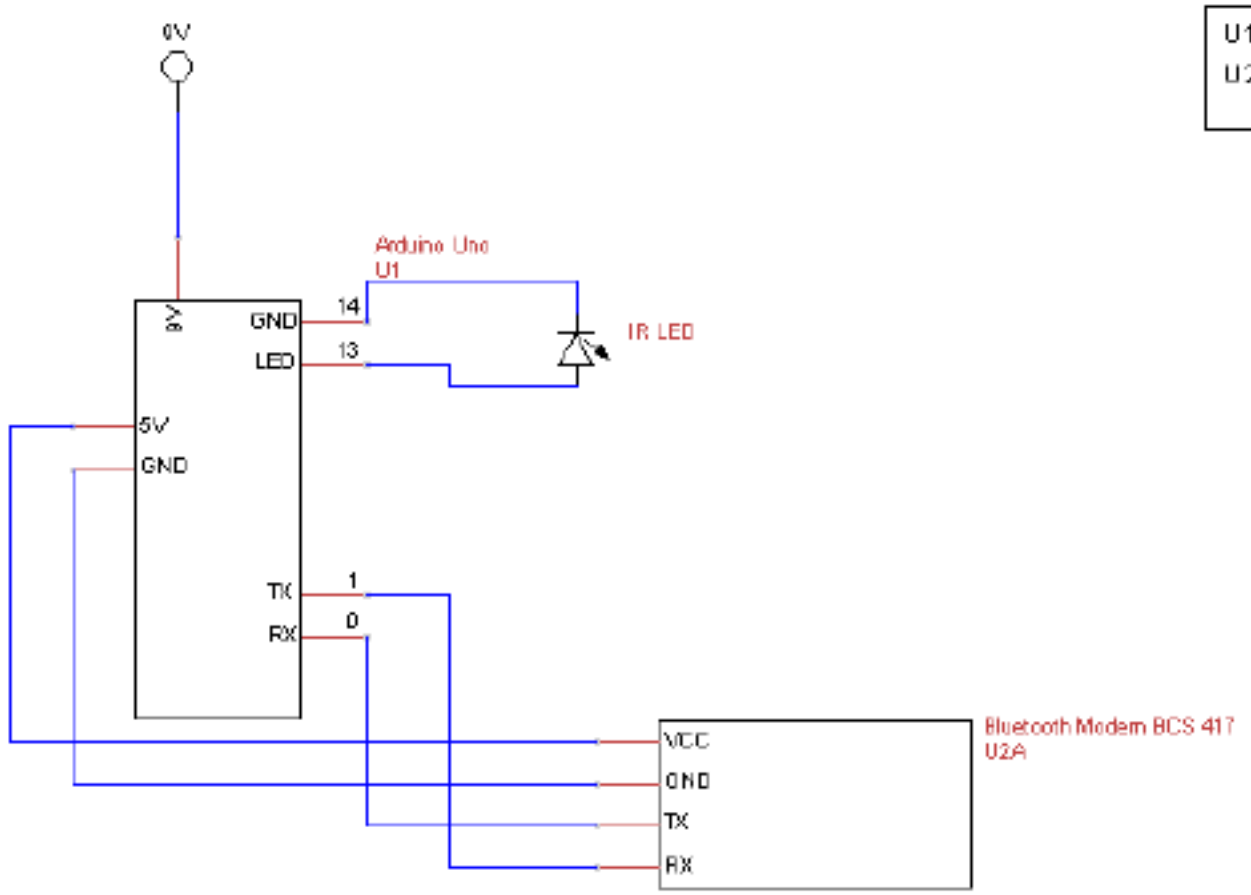
- There are many smart phone universal remote apps available, but they require the use of an IR dongle plugged into your phone to send a signal to the TV

Potential Applications

- Anyone with an android phone

Future Improvement Ideas

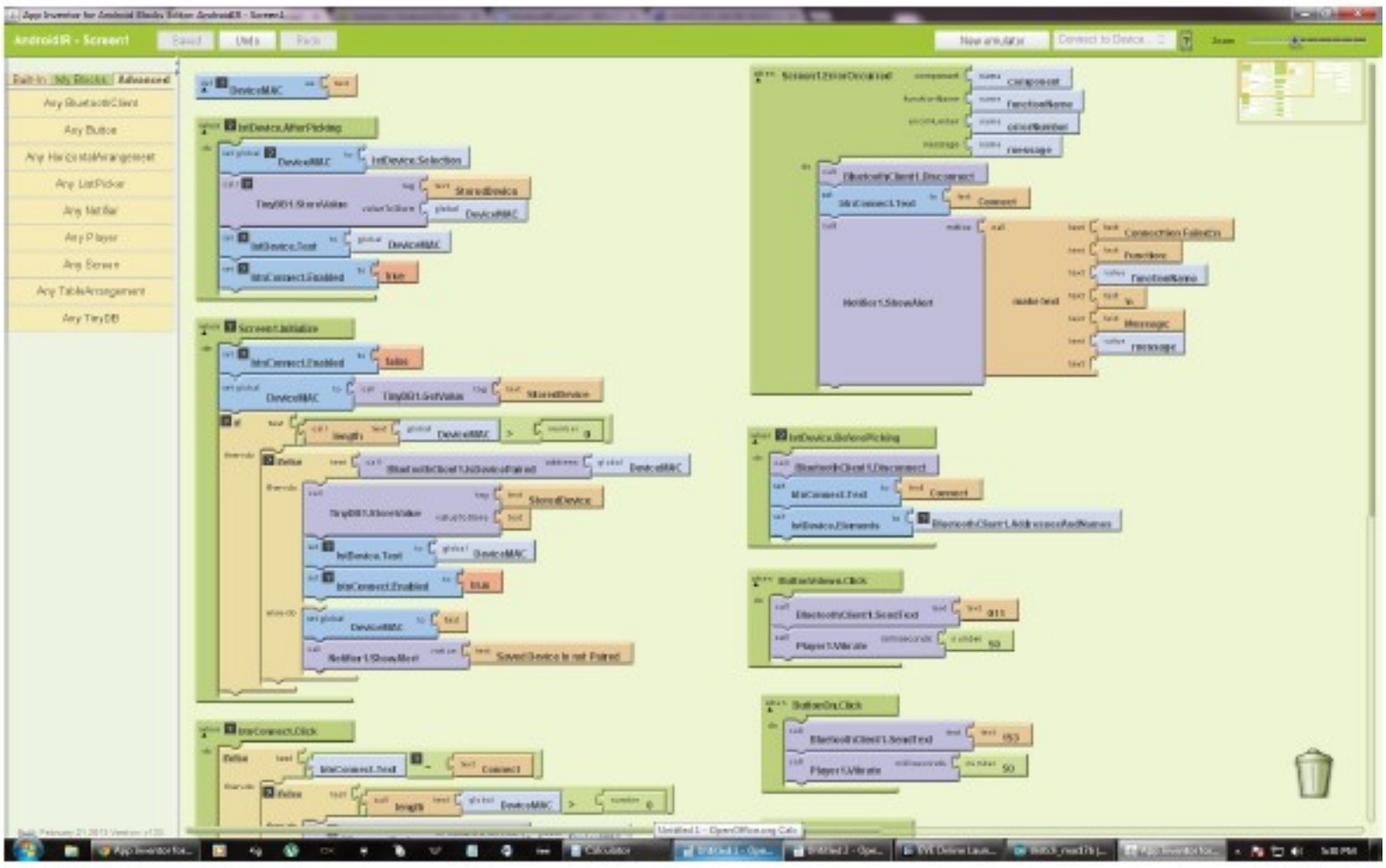
- Any device that recieves IR signals could be controlled
  - Robots
  - Remote controlled cars
  - An IR wall outlet would allow you to control anything that plugged into that socket

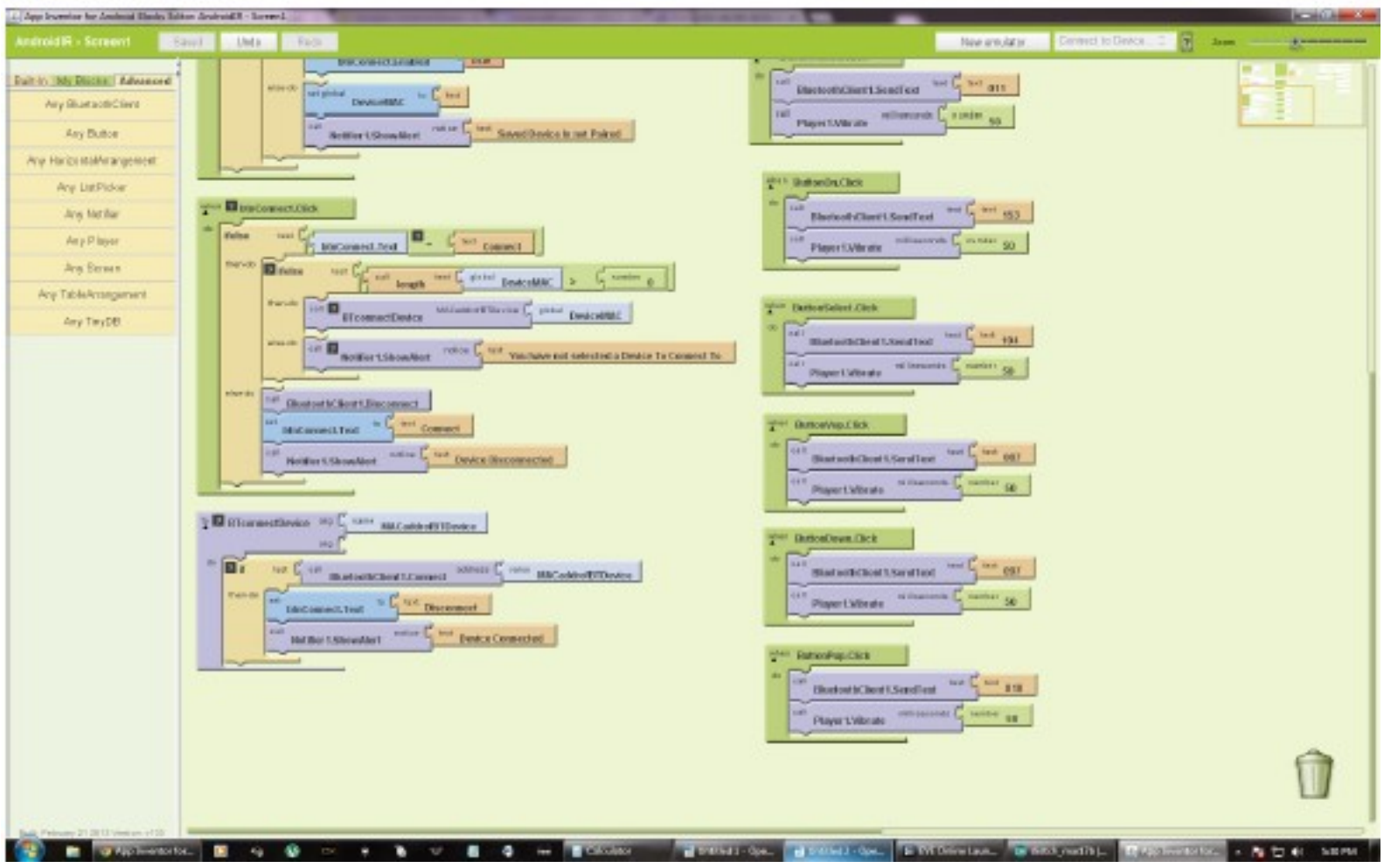


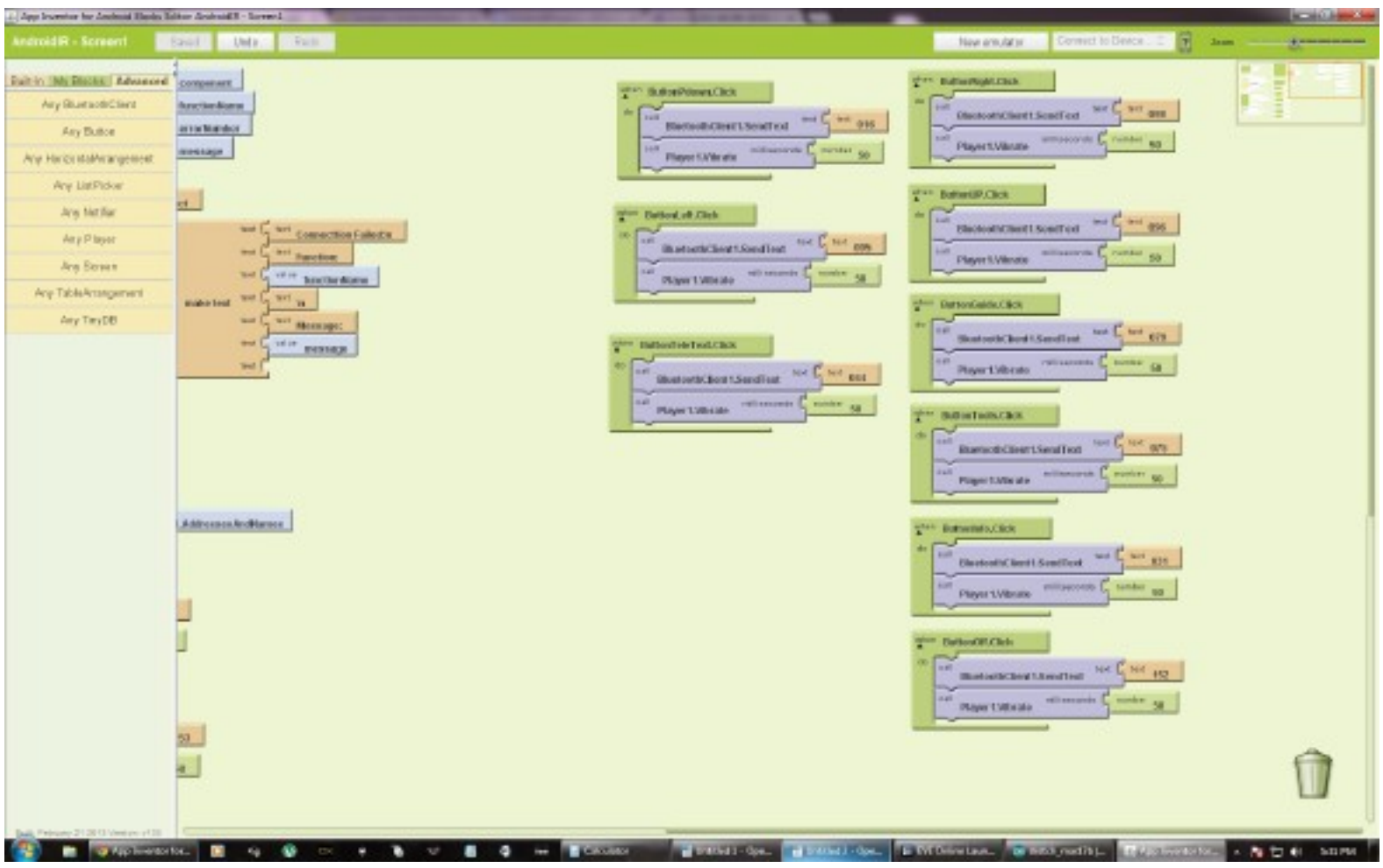
U1 Arduino Uno
U2A BCS 417

Title Bluetooth to IR translator		
Author Colton Wells, Tin Nguyen		
File		Document
Revision 1.0	Date	Sheets 1 of 1

App inventor blocks:






Android4R - Screen1
Built-in
My Blocks
Advanced
Any BluetoothClient
Any Button
Any HorizontalArrangement
Any ListPicker
Any Notifier
Any Player
Any Screen
Any TableArrangement
Any TinyDB
component
functionName
errorNumber
message
Connection Failed in
Function:
functionName
to
Message:
message
AddressesAndNames
Connect to Device ...
Zoom

Source for arduino board:

```
#include <util/delay.h>

#define IR_PIN 13

// With CONTINOUS defined, the first command is repeated continuously until
// you reset the Arduino.  Otherwise, it sends the code once, then waits for
// another command.
// #define CONTINUOUS

// Times are in microseconds
#define ON_START_TIME 4500
#define OFF_START_TIME 4500
#define ON_TIME 580
#define OFF_TIME_ONE 1670
#define OFF_TIME_ZERO 540

#define DEVICE_1 7
#define DEVICE_2 7
```

```
void setup() {
  pinMode(IR_PIN, OUTPUT);
  digitalWrite(IR_PIN, LOW);
  Serial.begin(9600);
  delay(1000);
  Serial.print("Starting up..\n");
}

byte command = 0;
int commandCount = 0;
bool commandReady = false;


void loop() {
  if (commandReady) {
    Serial.print("Writing command ");
    Serial.print(command, DEC);
    Serial.print("\n");

    writeStart();
    // Writing device code
    writeByte(DEVICE_1);
    writeByte(DEVICE_2);

    // Writing command code
    writeByte(command);
    writeByte(~command);
    writeEnd();
    delay(100);

#ifndef CONTINUOUS
    commandReady = false;
    command = 0;
    commandCount = 0;
#endif
    return;
  }
```

```
  if (Serial.available() > 0) {
    // Read in a 3-digit decimal command code.
    byte incoming = Serial.read();
    if (incoming <= '9' || incoming >= '0') {
      command *= 10;
      command += incoming - '0';
      ++commandCount;
    }
    if (commandCount == 3) {
      commandReady = true;
    }
  }
}

void writeStart() {
  modulate(ON_START_TIME);
  delayMicroseconds(OFF_START_TIME);
}

void writeEnd() {
  modulate(ON_TIME);
}

void writeByte(byte val) {
  // Starting with the LSB, write out the
  for (int i = 0x01; i & 0xFF; i <<= 1) {
    modulate(ON_TIME);
    if (val & i) {
      delayMicroseconds(OFF_TIME_ONE);
    } else {
      delayMicroseconds(OFF_TIME_ZERO);
    }
  }
}

void modulate(int time) {
  int count = time / 26;
  byte portb = PORTB;
  byte portbHigh = portb | 0x20; // Pin 13 is controlled by 0x20 on PORTB.
  byte portbLow = portb & ~0x20;
  for(int i = 0; i <= count; i++) {
      PORTB = portbHigh;
    _delay_loop_1(64);
    PORTB = portbLow;
    _delay_loop_1(64);
  }
  PORTB = portb;
}
```