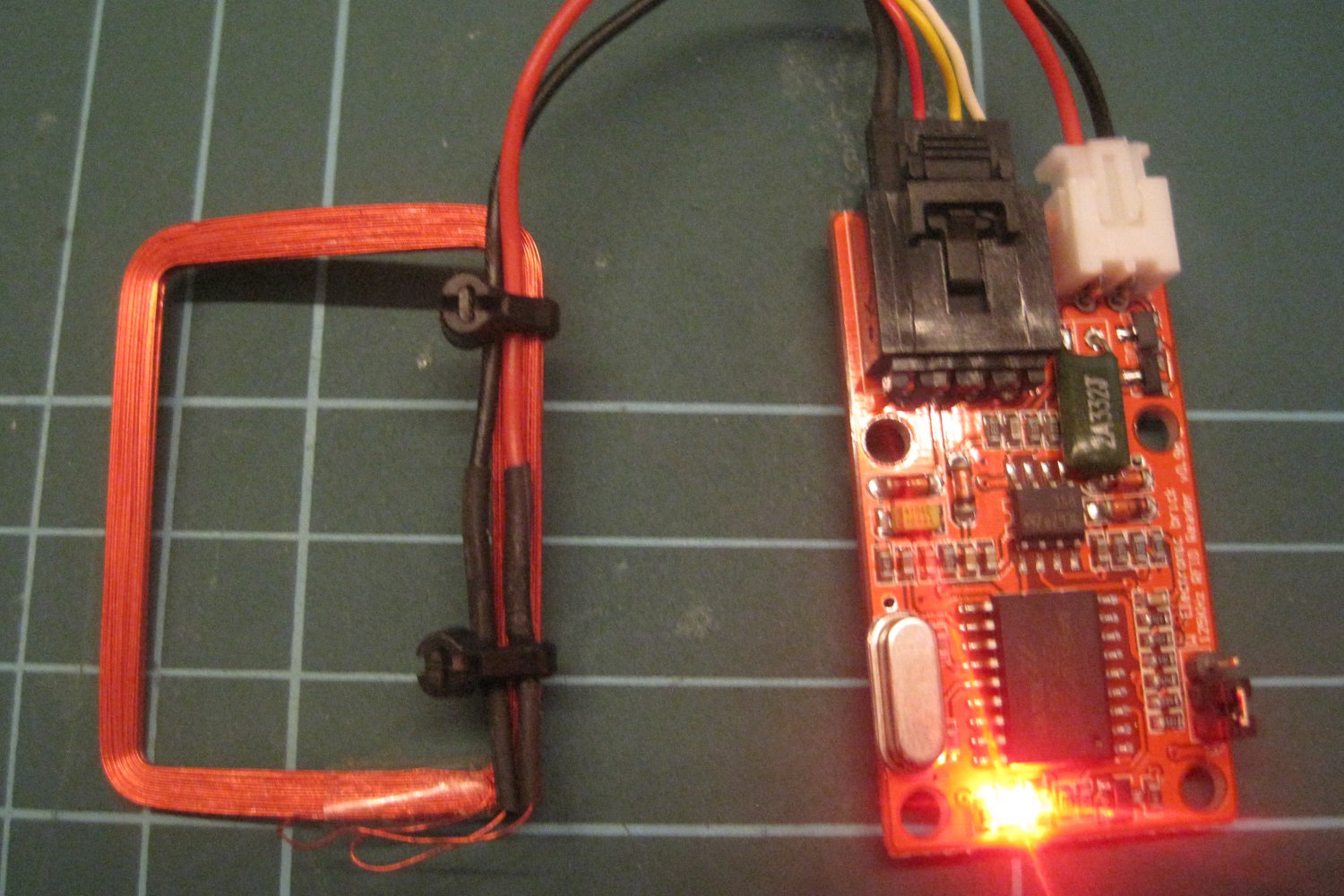
# BeagleBone and RFID

Quelle/Link: <http://mark.mckillen.com/2012/11/beaglebone-and-rfid/>

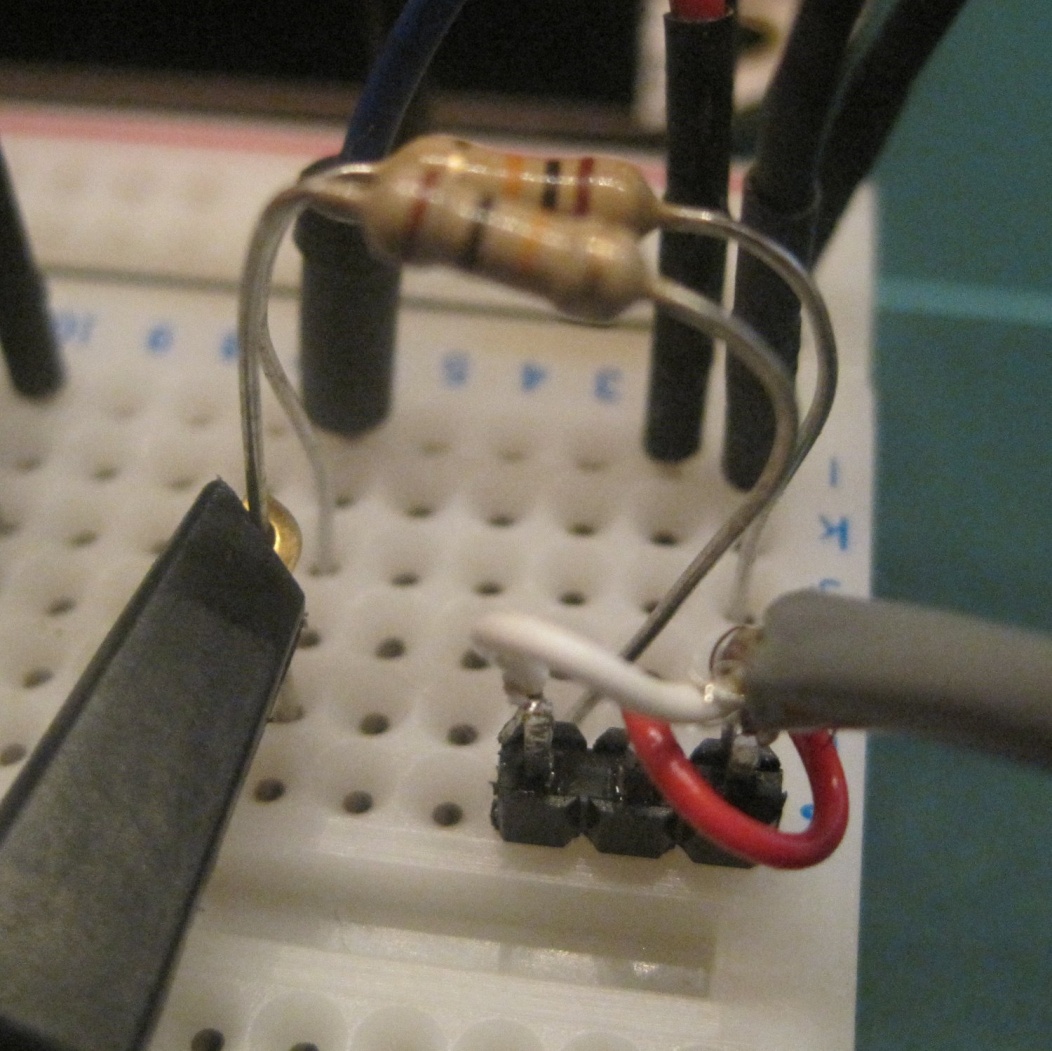
The BeagleBone has 6 UARTS on it.. (OK well 4 that are easy to use.) I thought it might be cool to look up a RFID reader I got a while ago. The reader caught my eye while I was ordering a few parts form [Watterott.com](http://www.watterott.com), the part is a [Electronic brick – 125Khz RFID Card Reader](http://www.watterott.com/en/Electronic-brick-125Khz-RFID-Card-Reader).  I had played about with it on my Arduino, but again without good ethernet connectivity I did not get too far.  Sure you could make a selfcontained door entry system, but a networked one give one many more options!

[](http://mark.mckillen.com/wp-content/uploads/2012/11/Electronic-brick-125Khz-RFID-Card-Reader.jpg)

Electronic brick – 125Khz RFID Card Reader

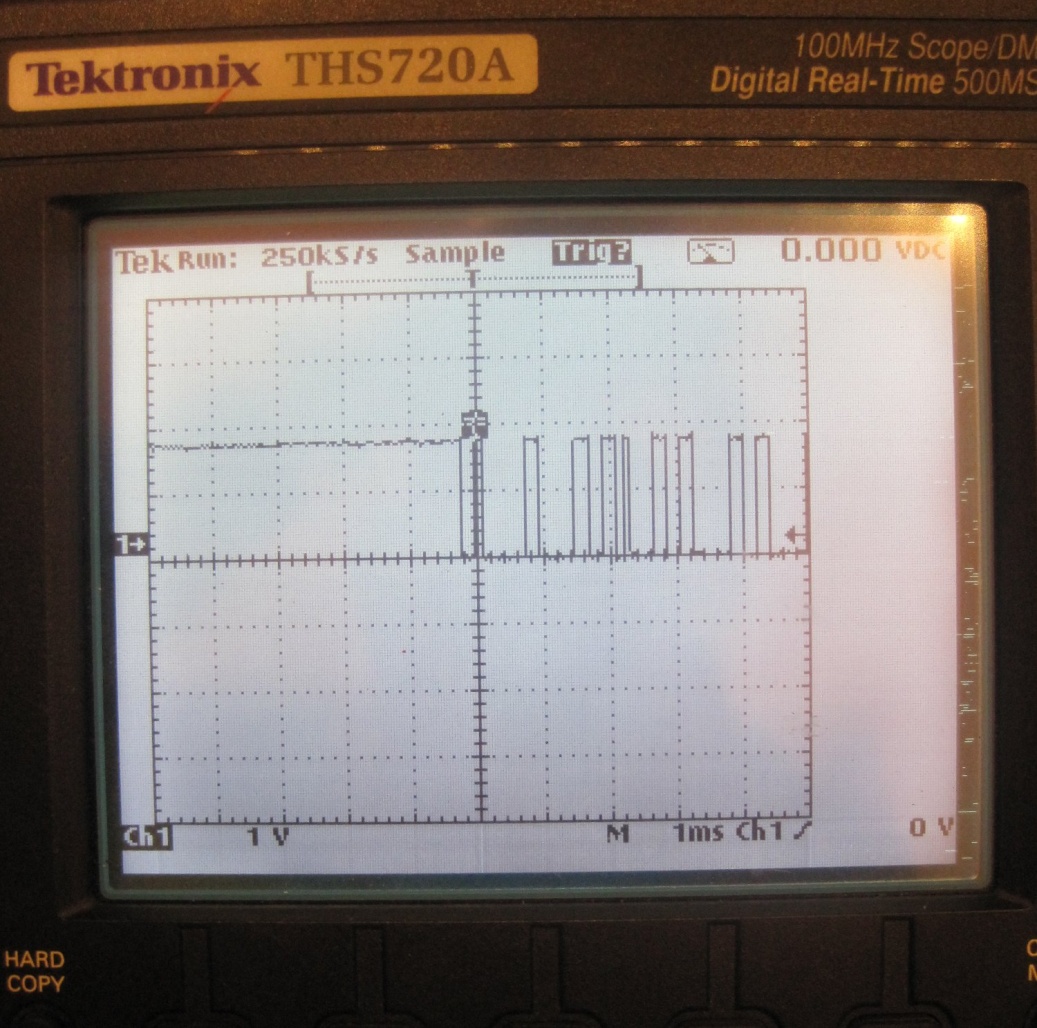
Back into the world if pin decoding and mux’ing on the BeagleBone.. A quick search found this great page with all the info I needed: <http://www.gigamegablog.com/2012/01/22/beaglebone-coding-101-using-the-serial-and-analog-pins/>

The RFID module works off 5V… Ummm need to do something about the levels! As the “brick” only outputs data I only have to worry about level shifting from 5 to 3.3V.  What could be easier? A simple resistor based voltage divider should do it the trick. A 10K from the output of the RFID module, to the UART1\_RXD pin (P9\_26) and another resistor from P9\_26 to ground.

[](http://mark.mckillen.com/wp-content/uploads/2012/11/Probe-on-voltage-divider.jpg)

The wires above are: White – Data, Red – +5V, Black – 0V (hidden from view!)

Check the levels on the scope…

[](http://mark.mckillen.com/wp-content/uploads/2012/11/RFID-serial-data-on-Tek-DSO.jpg)

Vertical scale is at 1V/div… so we are getting 2.5V or there abouts.  Should be OK to drive the UART on the BeagelBone.

On my Ubuntu distro I had to install the python-serial package before running this code. Now time for some Python:

import serial, os

import sys

# -------------- configurable settings ---------

PORT = '/dev/ttyO1'     # set tty port

                        # NOTE: ON BEAGLE BONE O1 is the Letter O

BAUD\_RATE = 9600        # set baud rate

RX\_PIN = 'uart1\_rxd'    # port pin for RX

MUX\_MODE = 0            # MUX mode for PIN

MODE\_EXT = 32           # Enable receive on pin too!

open('/sys/kernel/debug/omap\_mux/' + RX\_PIN, 'wb').write("%X" % (MODE\_EXT + MUX\_MODE))

ser = serial.Serial(PORT, BAUD\_RATE)   # open serial port

have\_tag\_data = 0

while True:

        a = ser.read()                 # read byte

        if ord(a) == 3:

                print "Tag stop!"

                have\_tag\_data = 0

        if ord(a) == 2:

                print "Tag start!"

                have\_tag\_data = 1

        if have\_tag\_data>1:

                print (have\_tag\_data-1), ord(a)         # print byte and count

                have\_tag\_data += 1

        if have\_tag\_data==1:

                have\_tag\_data += 1

Run it and watch the numbers change for different tags:

root@bbone1# python rfidtest.py

Tag start!

1 51

2 68

3 48

4 48

5 52

6 55

7 53

8 56

9 66

10 53

11 57

12 55

Tag stop!

Tag start!

1 48

2 49

3 48

4 48

5 48

6 66

7 57

8 52

9 66

10 55

11 50

12 57

Tag stop!

So 2.5V can drive the UART no problems… All good!