Micromouse Registration and Timing System

RATS

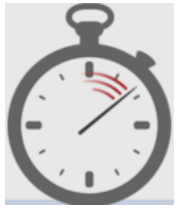
David Hannaford &
Ian Butterworth

MINOS 2018

A two part presentation

- David Hannaford
  - Objectives
  - Functionality
  - Timing hardware and design
- Ian Butterworth
  - Display and registration components
  - PC Interfacing protocol
  - Data model
  - Demos

Core Functionality Required



- Measure the time between a mouse leaving the start cell and arriving in the centre cell

- Store and display times for all the runs for a mouse
- Show leader board for all mice in a competition

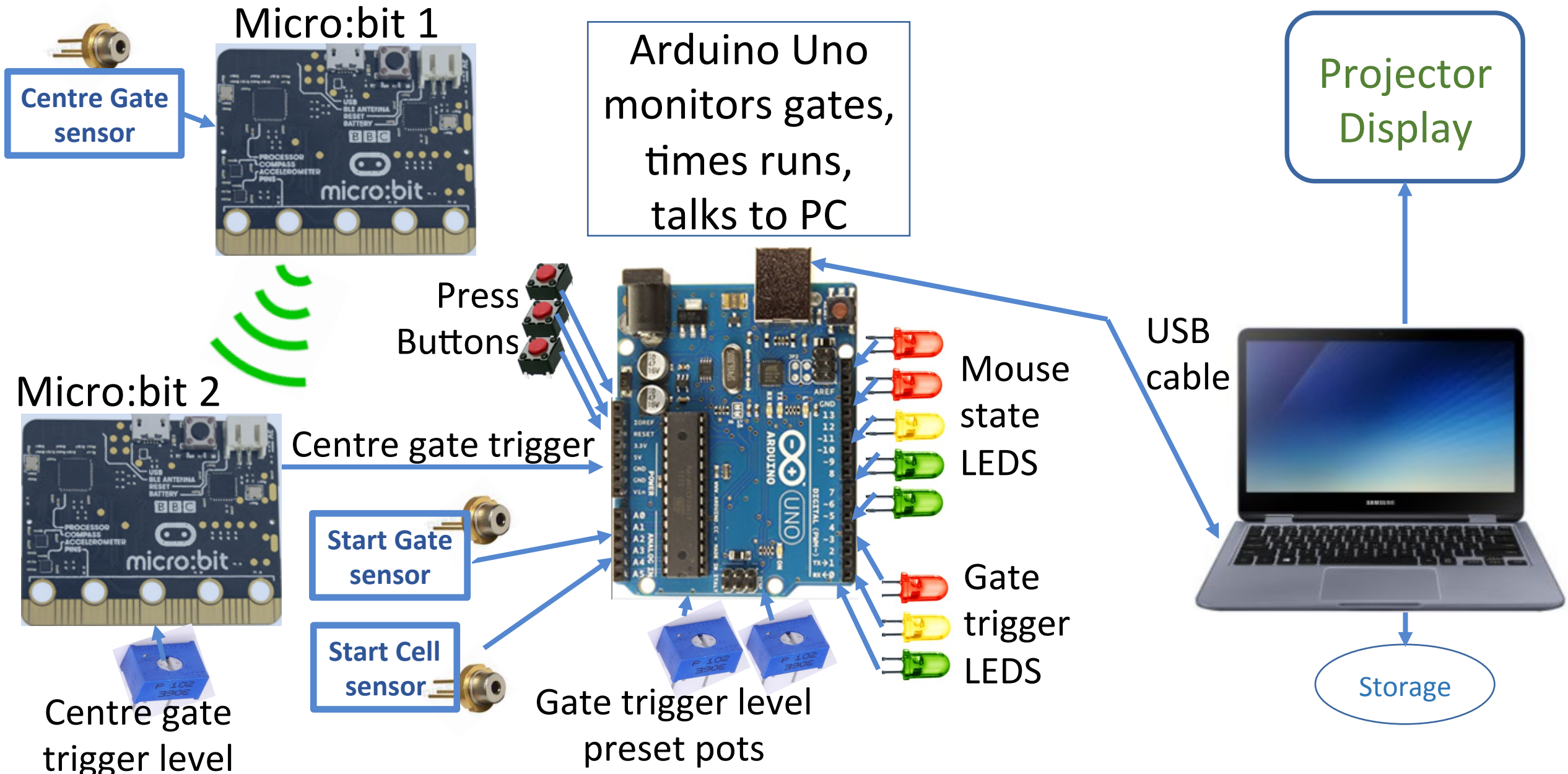
- Minimal operator actions – so automatically detect start and end of a run and handle aborted runs

- To have no wires going underneath the maze

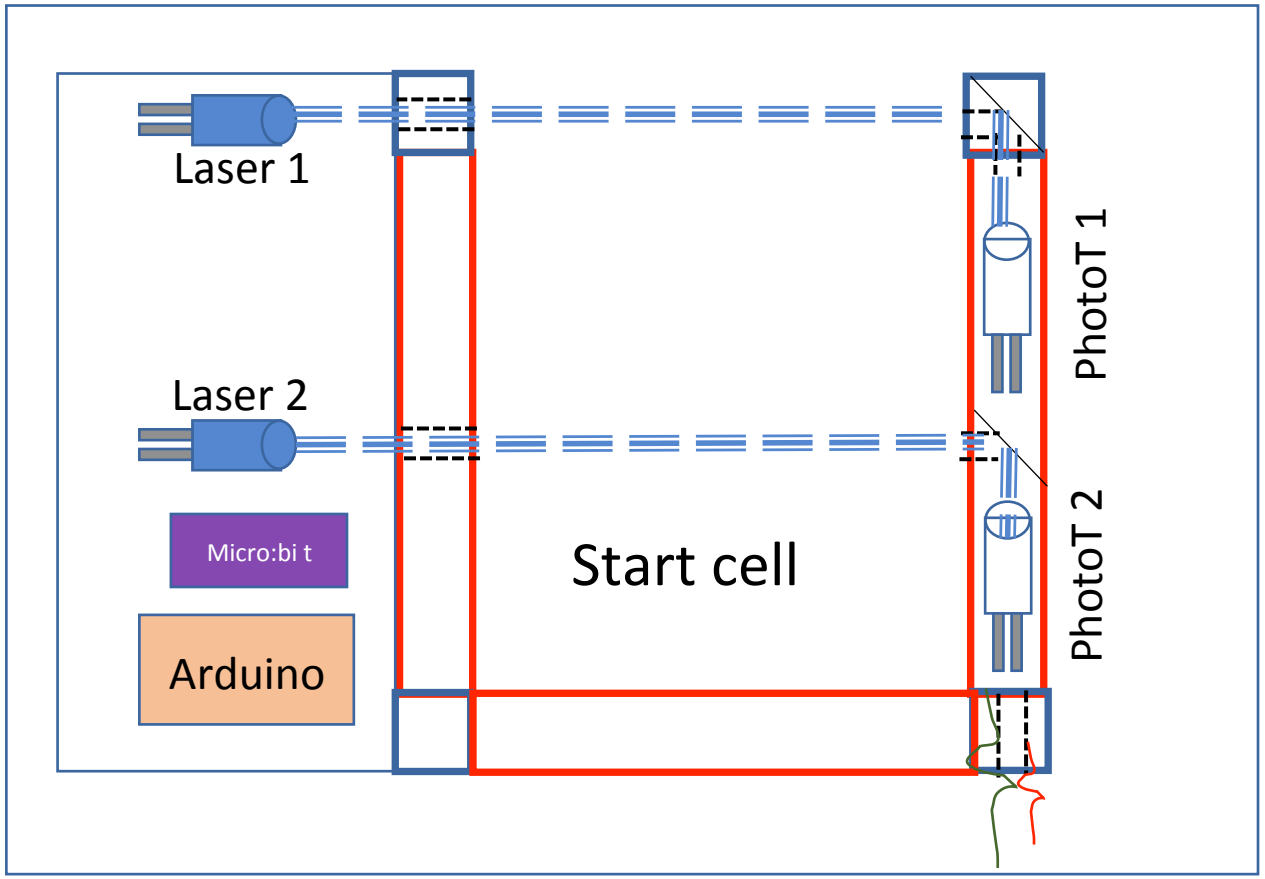
Design Requirements

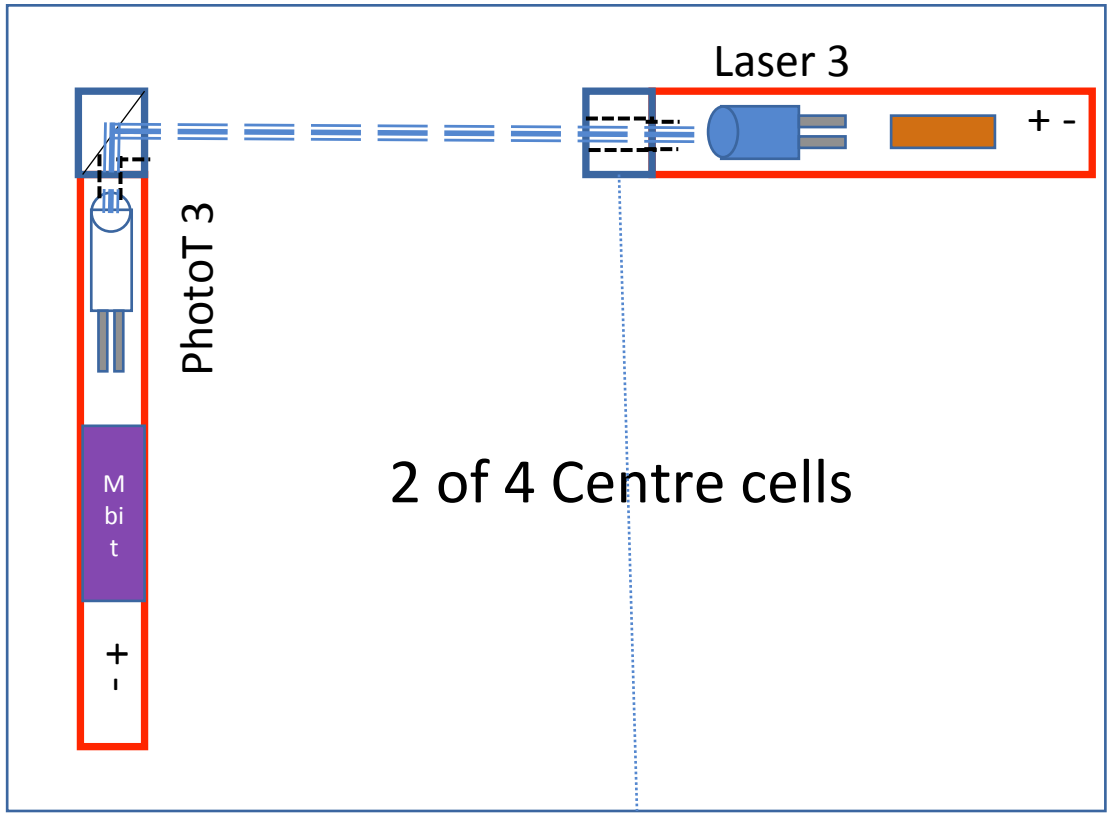
- Use solution components already familiar with, in our possession or readily available to both of us
- Split between measuring part of solution and the PC based storage and display part so that they could be developed separately in parallel – with agreed protocol between the 2 parts
- Centre gate sensing using battery power and wireless communication so as to avoid wires under the maze
- Able to detect mouse being picked up and put back in start cell with or without reaching the centre to start a new run
- Sensing components, batteries and communication hardware all have to fit within the available plastic walls (8mm max width of any part)

Block diagram of main hardware components



Start cell and centre gate layouts





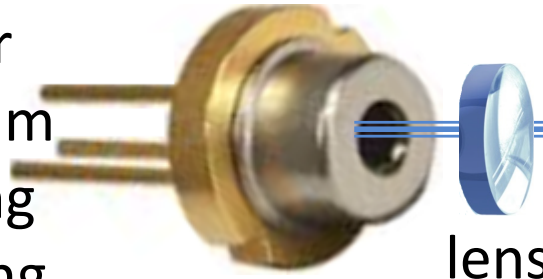
Why 3 Laser beams?

- With just a beam at the start gate and at the centre gate you can't tell whether a mouse is coming back to the start cell, or leaving it after being picked up and put back at the start.
- By having a start cell sensor as well as the start gate the software can detect when a mouse is ready to start a new run regardless of how it got to the start cell – thereby avoiding manual intervention between runs
- With 3 sensors a 5 state machine can identify where the mouse is in the maze in respect of the timing needs – and indicate this on the state LEDs

- 1 Mouse in start cell waiting to start a new run, cancel any current timings in progress
- 2 Broken the start beam, so we should start the timer
- 3 Fully left the start cell, mouse running in maze
- 4 Broken centre gate beam, so stop timer and pass data to PC
- 5 In centre cells, ignore gate sensors until back in start cell

The Gate sensors

3.3v Regulated supply to laser through 15 ohm current limiting resistor running at 90mA



445 nm blue laser diode

lens

Photo transistor BPW85C

3.3V

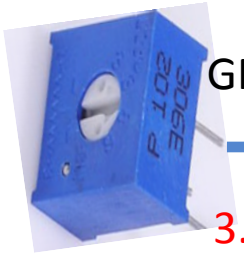
3.9kΩ

GND

ADC 1

Arduino compares phototransistor value with potentiometer value and indicates if gate is triggered

ADC 2



GND

3.3V

Laser diode bought on Ebay with lens for about £6. It probably comes out of blu-ray DVD players – needs a heat sink

Blue laser chosen because:

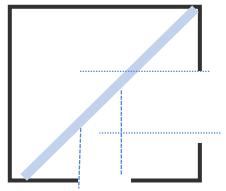
- Less likely to interfere with intra-red sensors on mice than a red or infra-red beam
- Smaller beam than from an LED so again less likely to affect mouse sensors
- Able to see blue laser spot, so it is easier to set it up pointing directly at phototransistor

The modified posts

Type 1

Straight through 3mm hole for laser beam to shine through at 1cm from ground

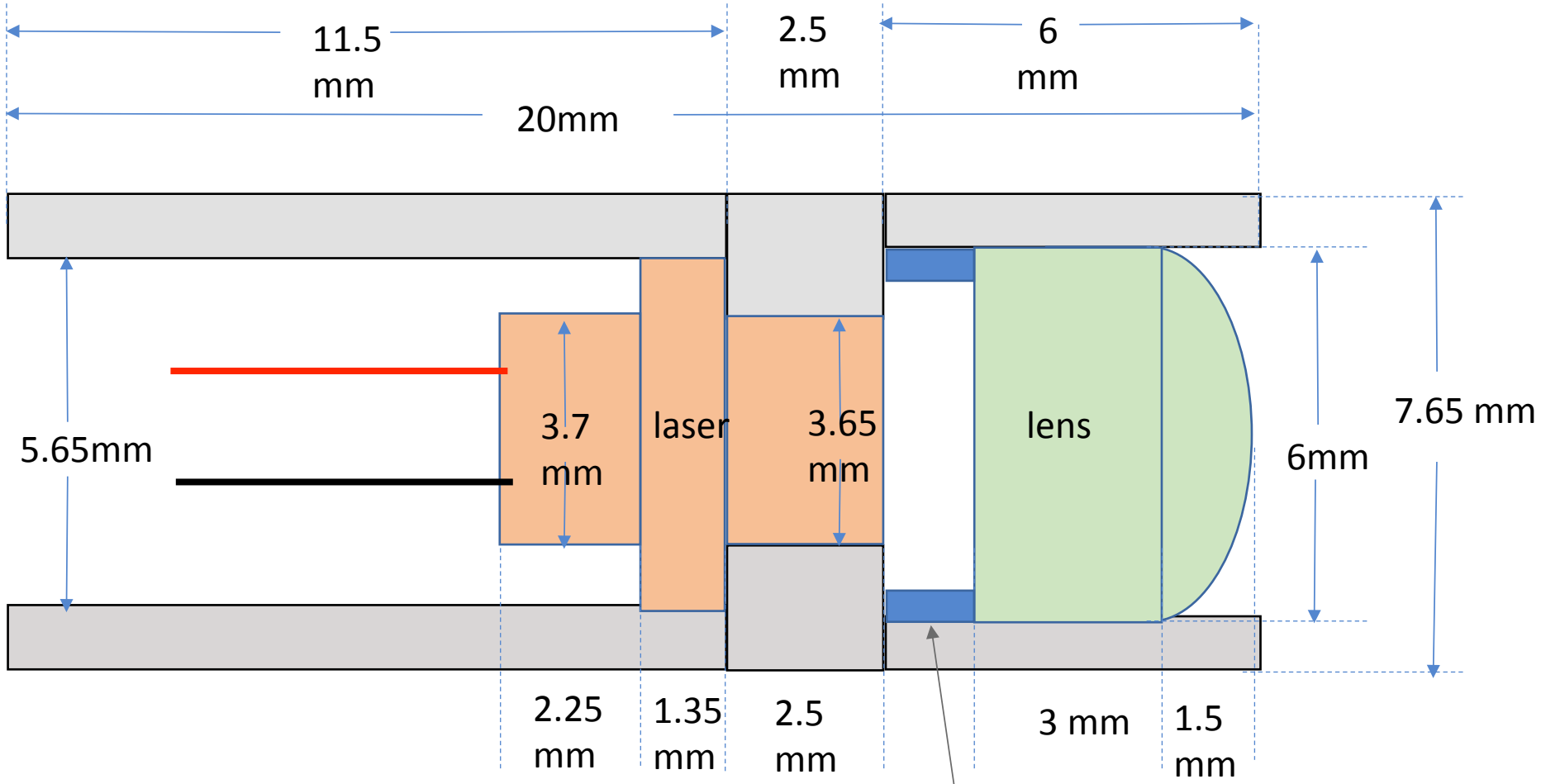
Type 2



Right angle reflector with two 3mm holes on adjacent sides with foil reflector at 45 degrees inside the post for laser beam to reflect 90 degrees onto photo-transistor

By putting the reflector posts in at different orientations we can cope with the centre gate being in any position in the middle 4 cells

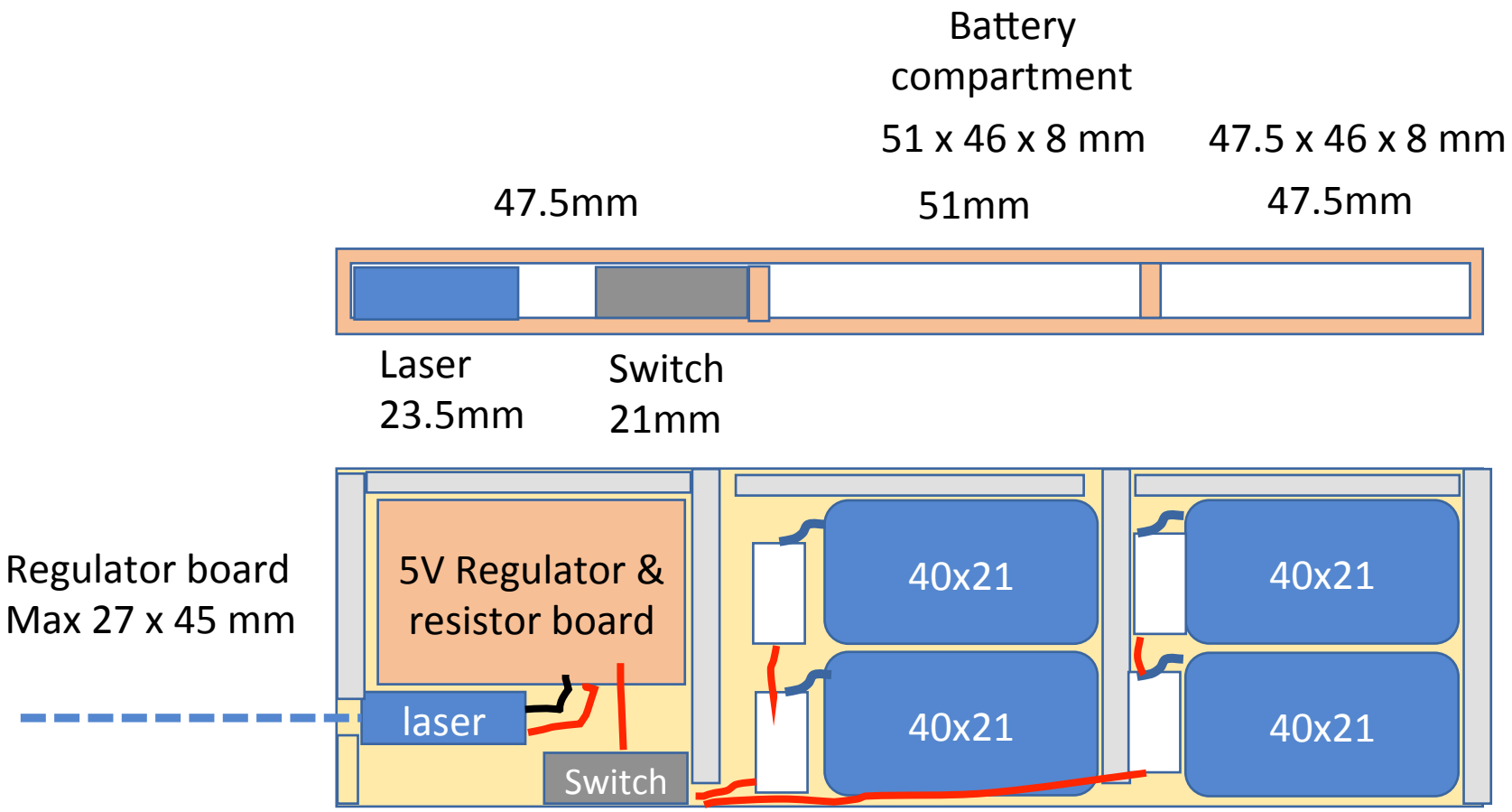
Blue laser heatsink and lens mount for inside wall



Heatsink made on lathe from piece of 10mm aluminium rod

Focussing shim
About 1.4 mm

Wall compartments for centre gate laser sender

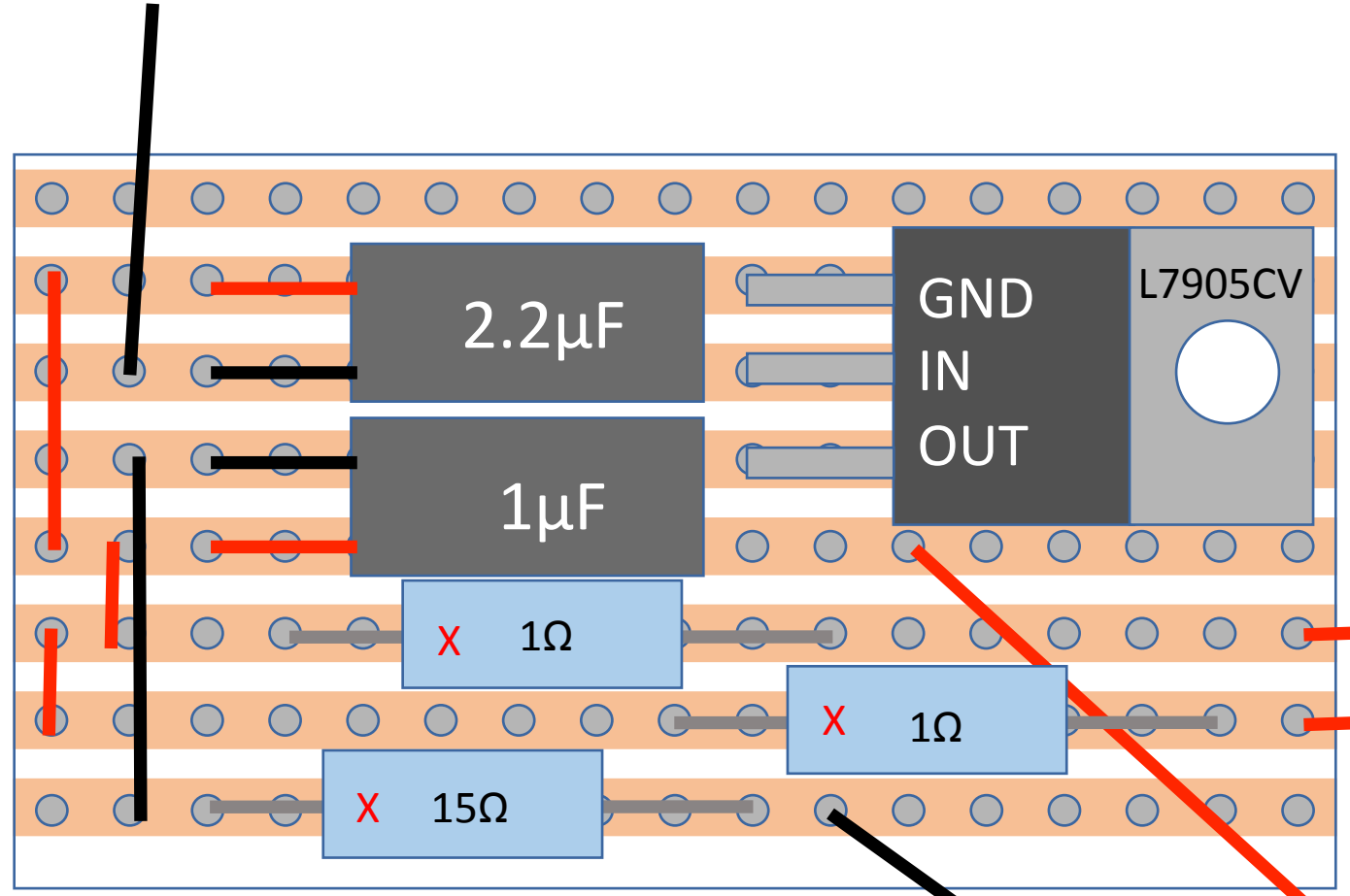


Batteries 4 x 400mAH 3.7V LIPO of size 40 x 21 x 7.5mm
Connected to give 7.2V 800mAH

Regulator board for laser drive

With 6.9 or greater volts in it gives 90mA through the laser

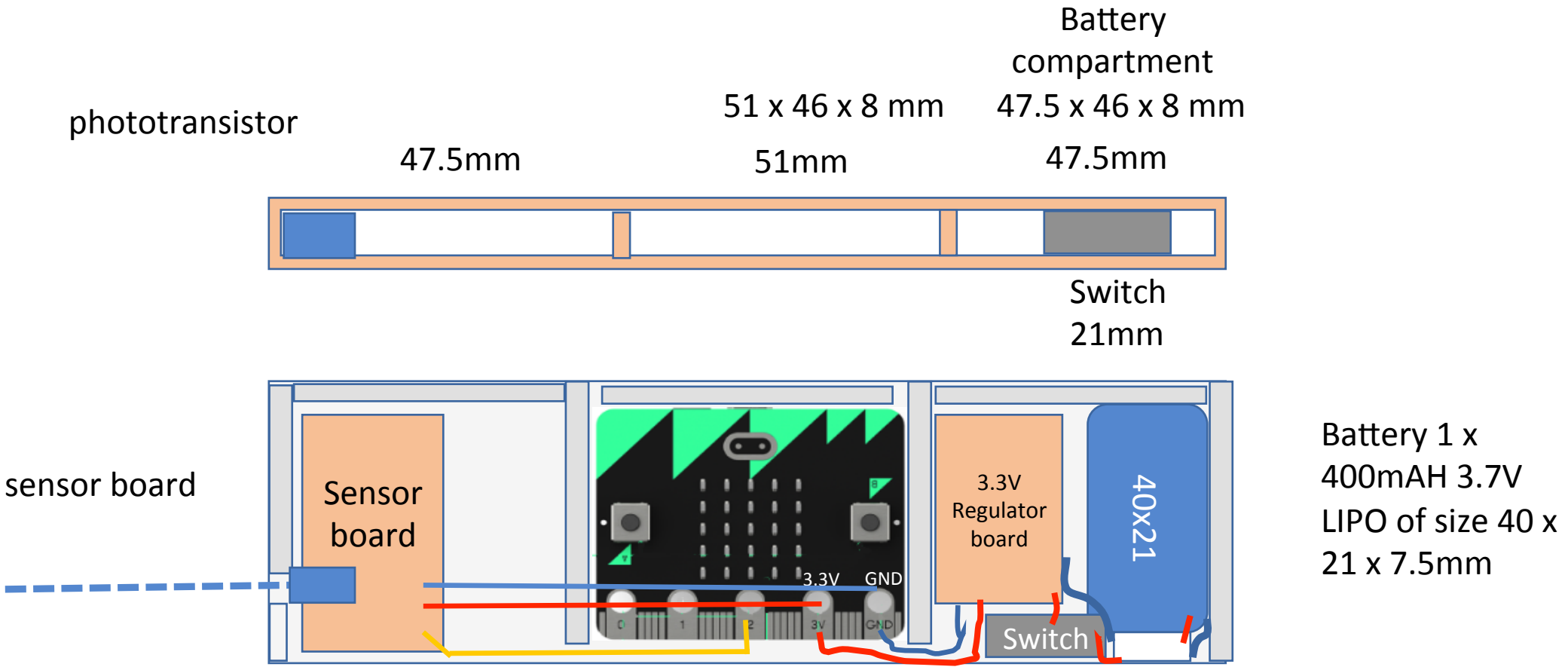
To scale at 4 times actual size



Note L7905CV negative 5V regulator used as happened to have it handy

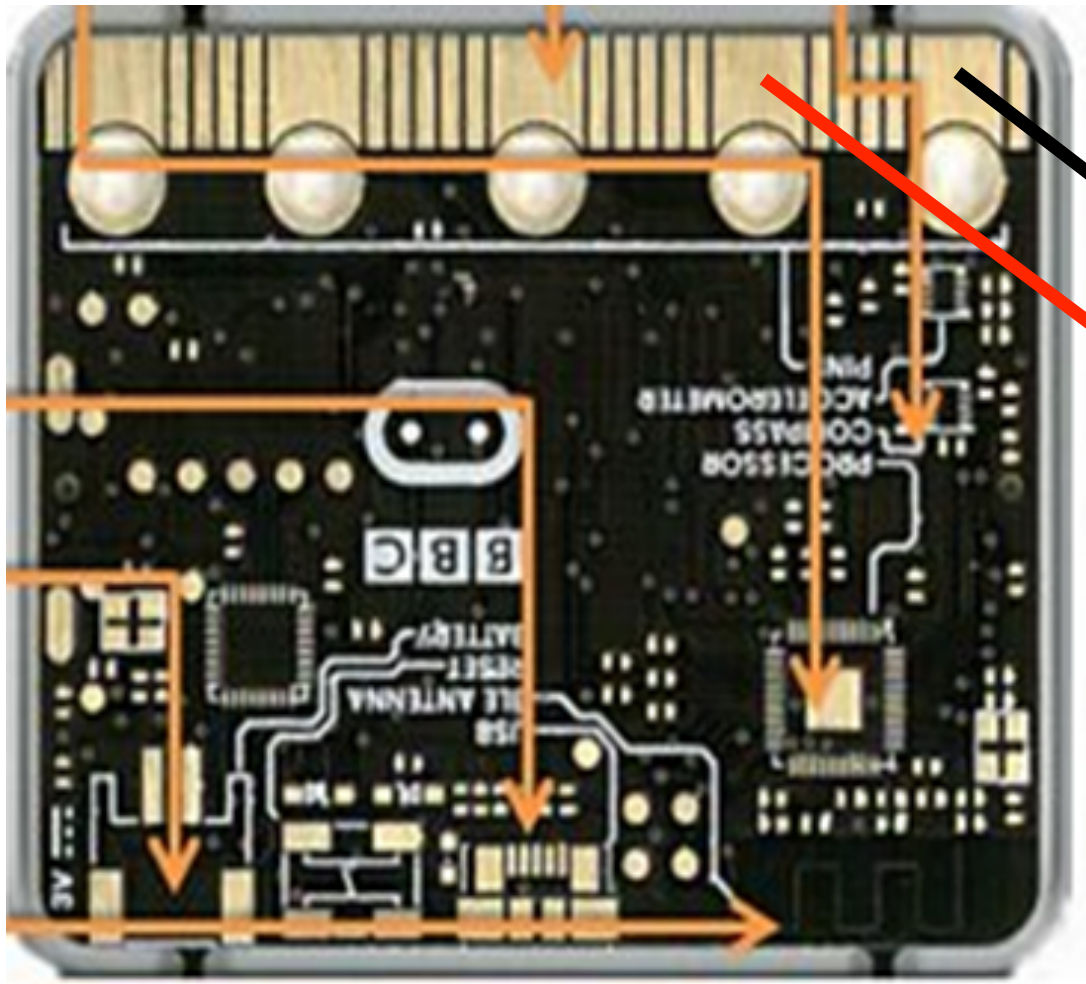
X = track cut below component

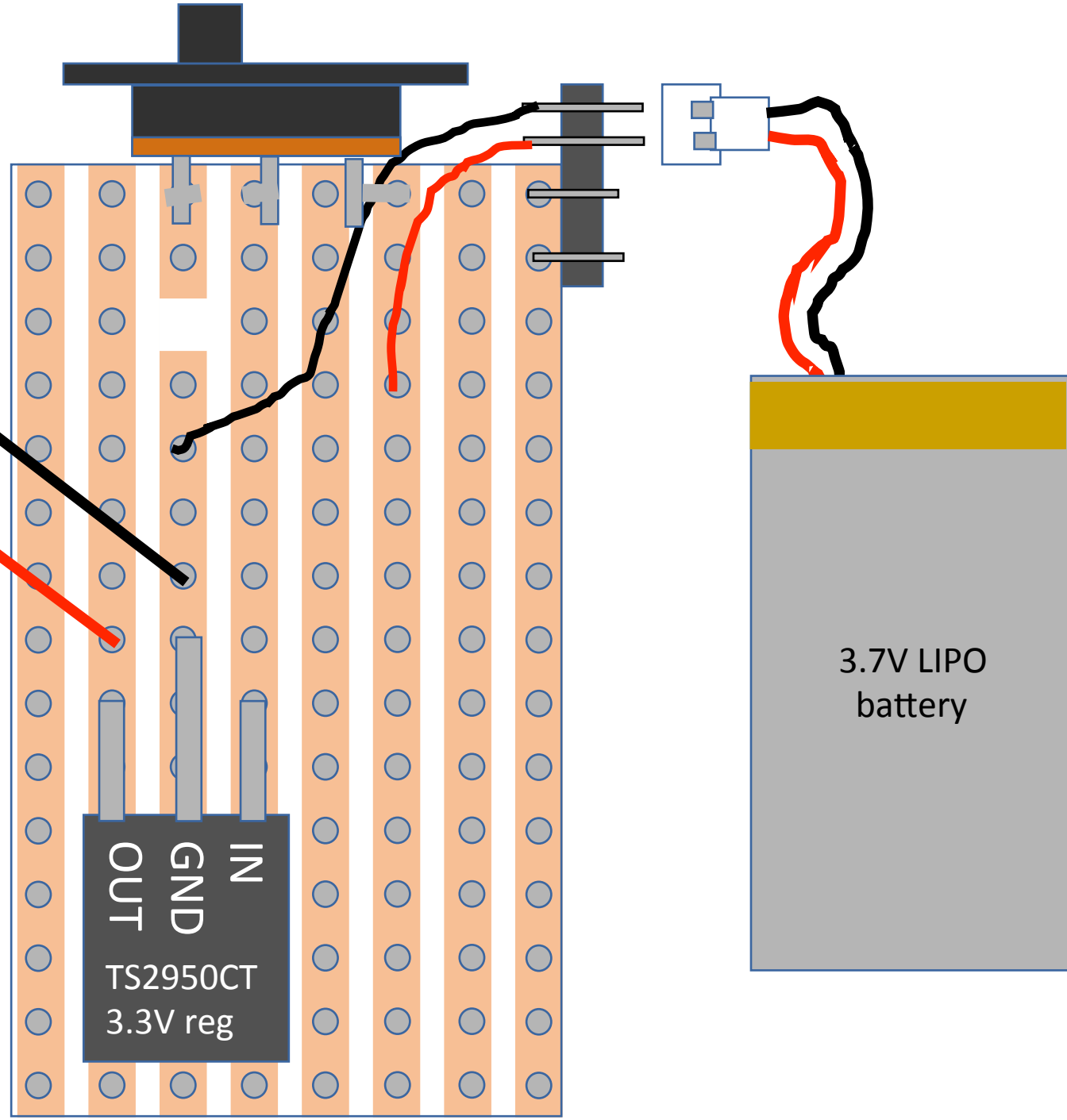
Centre gate wall compartments for photo detector and Micro:bit

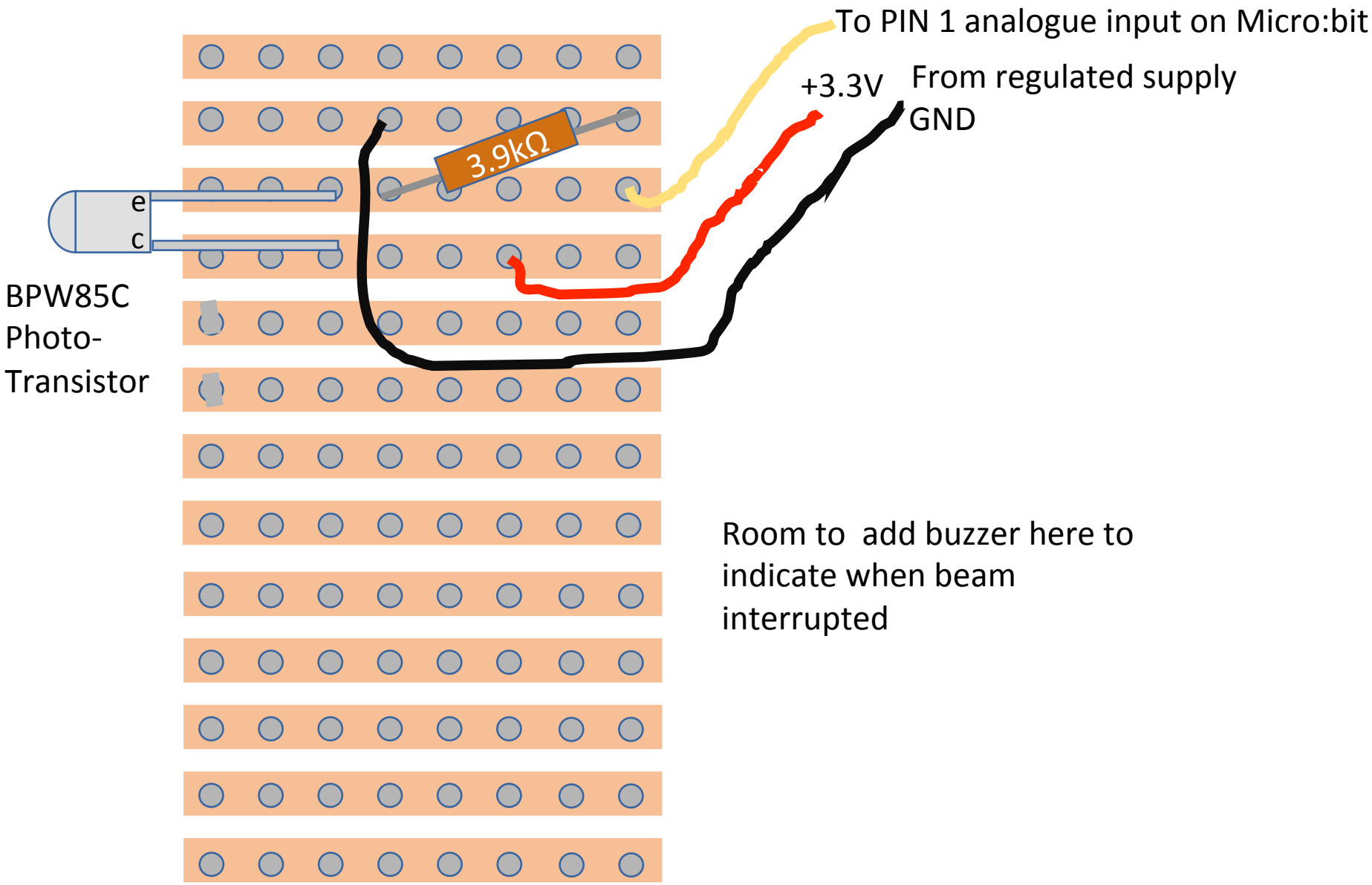


Battery 1 x 400mAH 3.7V LIPO of size 40 x 21 x 7.5mm

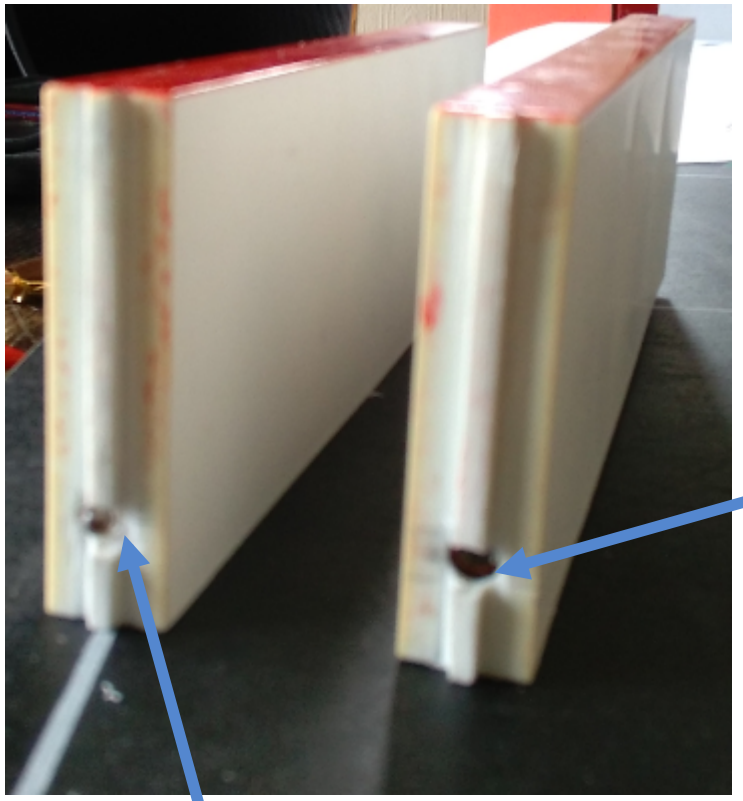
Power supply and switch for micro:bit in centre wall beam detector

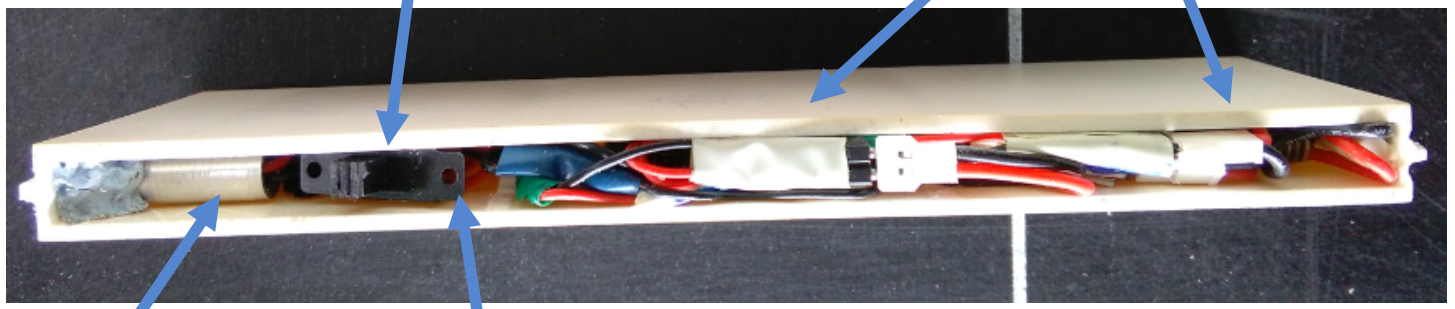


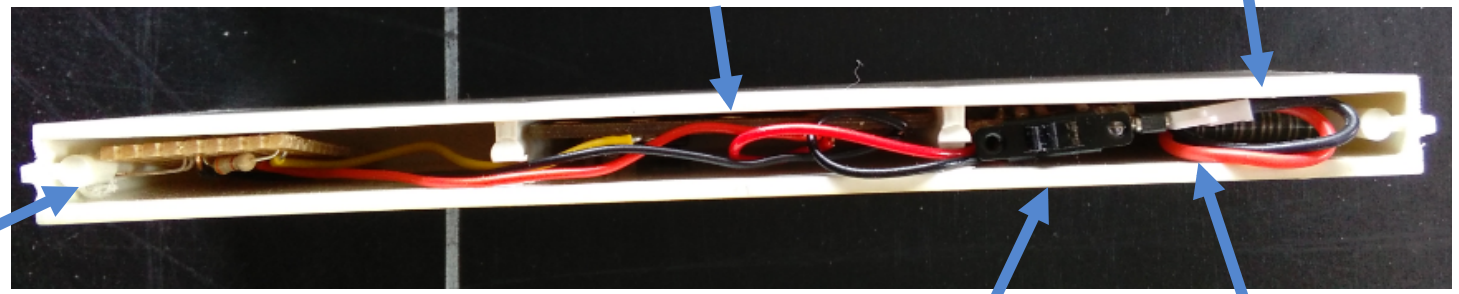



To PIN 1 analogue input on Micro:bit
+3.3V
From regulated supply
GND
3.9kΩ
e
c
BPW85C
Photo-
Transistor
Room to add buzzer here to
indicate when beam
interrupted

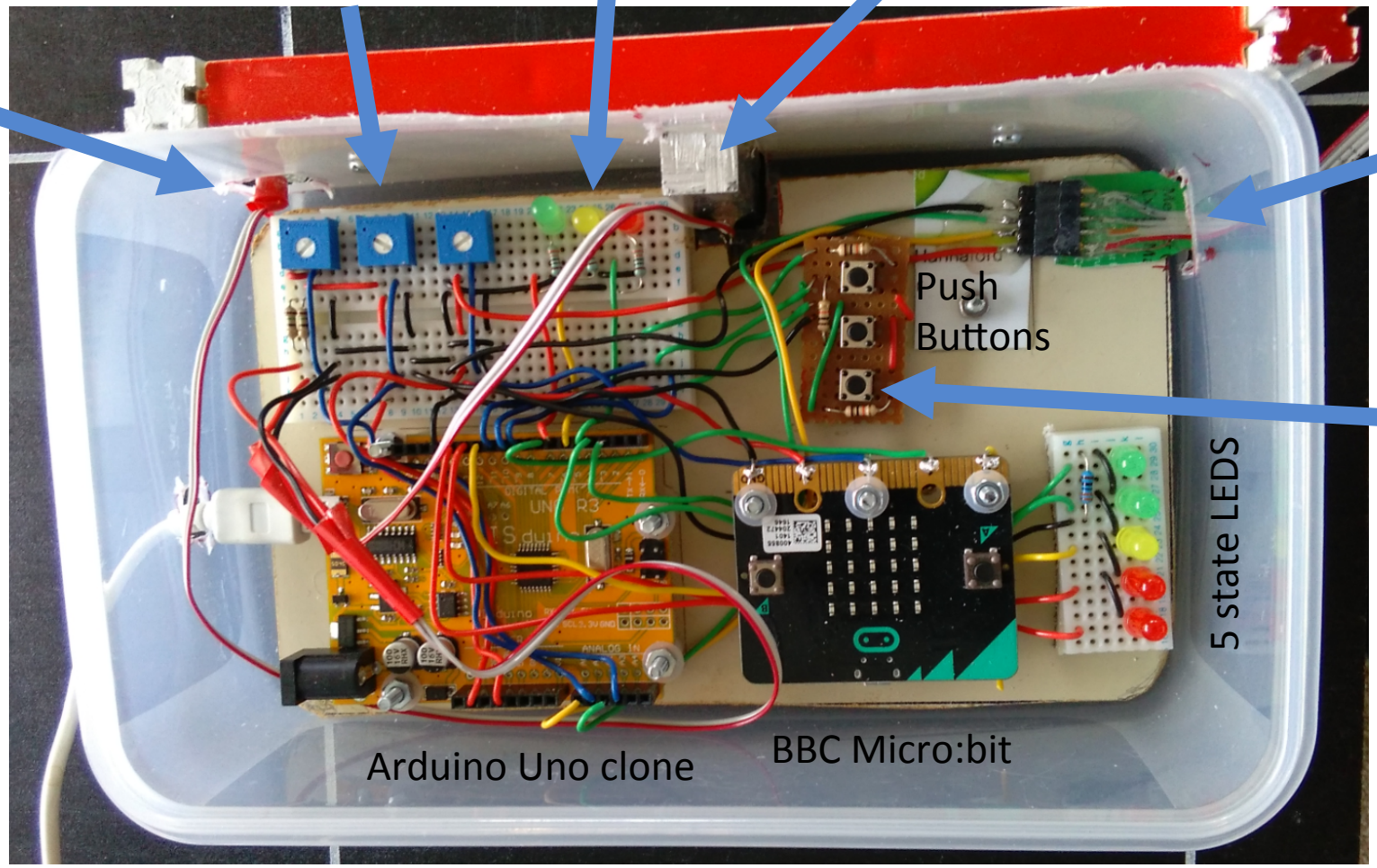
Physical implementation of centre gate walls







Physical implementation of timing hardware



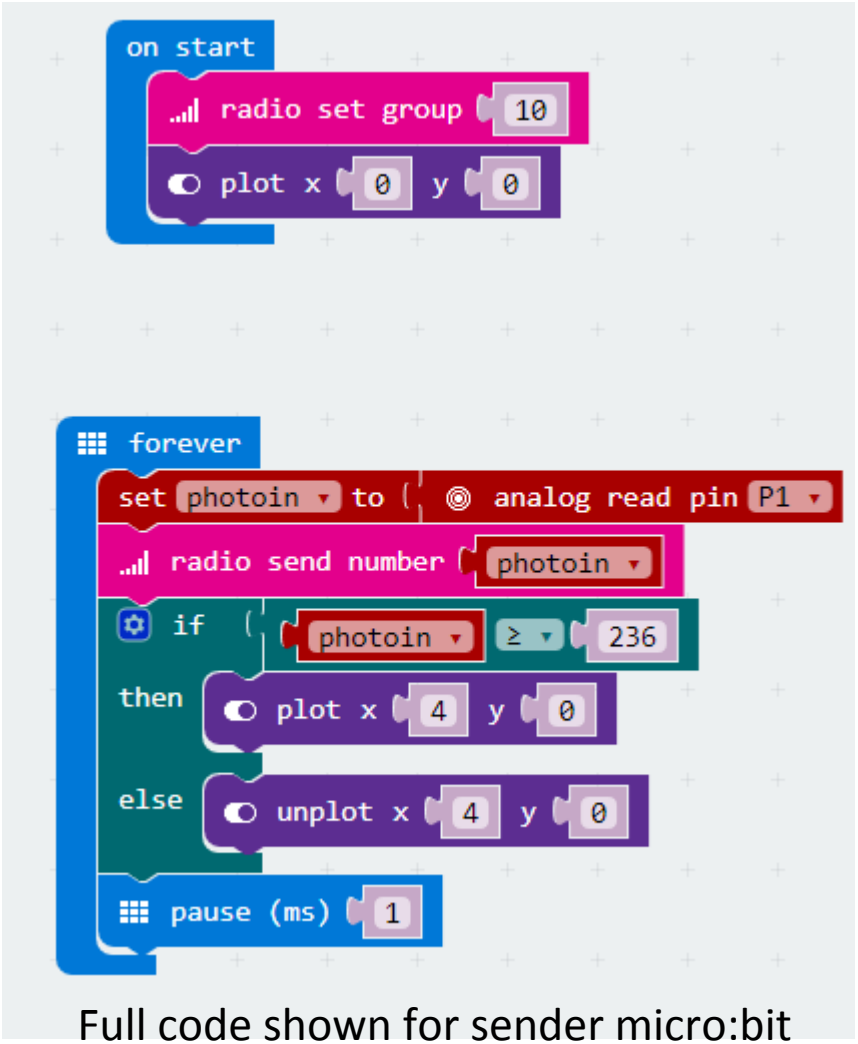
Gate trigger indicator LEDS

- Centre gate
- Start cell
- Start gate

Coding in timer components

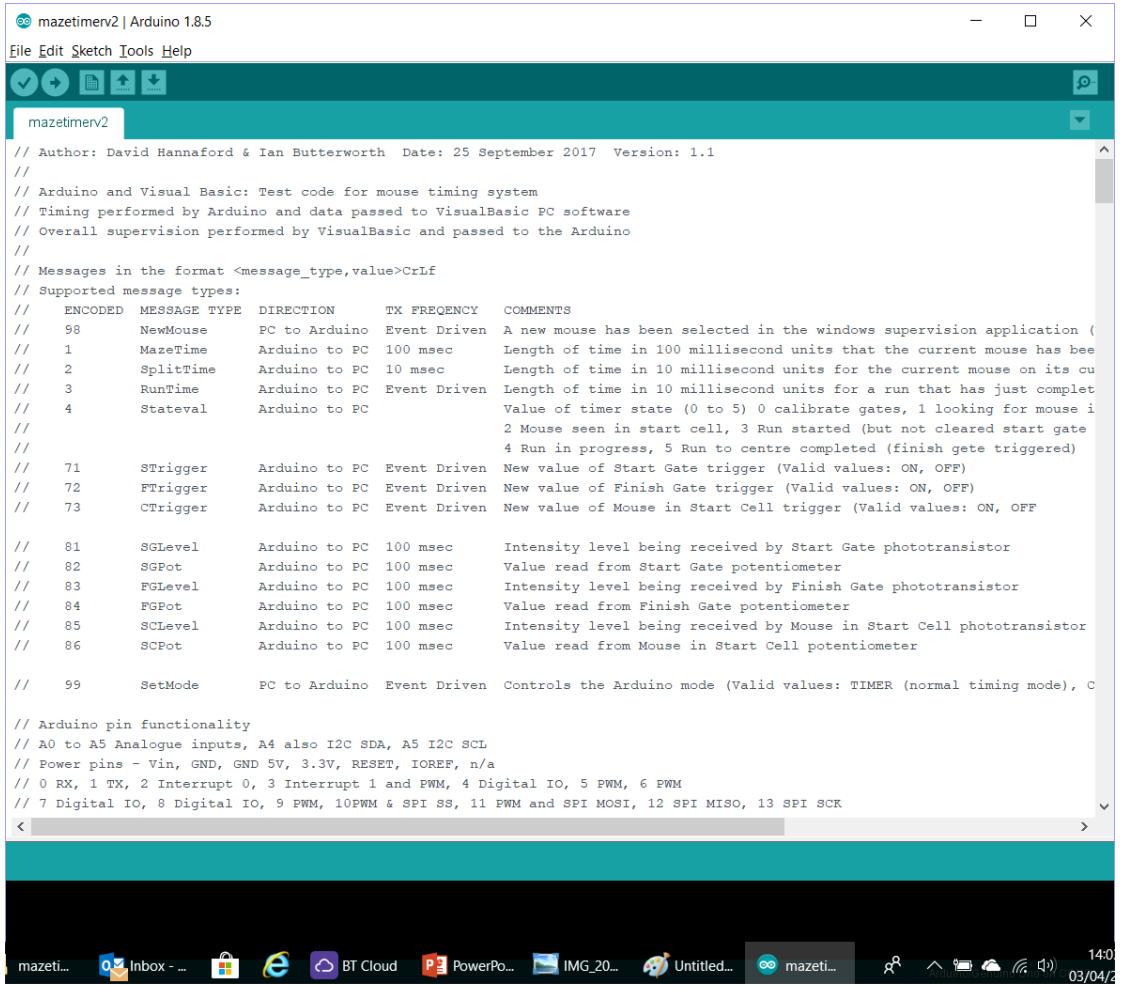
Minimal code in micro:bits

Block code in micro:bit



Full code shown for sender micro:bit

Sketch code in Arduino



About 500 lines of code in Arduino

State machine main logic

- 0 calibration mode at switch on or reset. Show gate trigger LEDs
  - When press button hit go to state 1
- 1 Looking for new mouse or mouse on way back to start cell after a run
  - Mouse seen in start cell go to state 2
- 2 Mouse in start cell, start beam not broken
  - Look for start beam to break. At break, send start time to PC and go to state 3
- 3 Mouse breaking start beam
  - Waiting to clear start gate. Update run number. Send split time to PC.
  - When clear of start gate go to state 4
- 4 Mouse clear of start gate and running in maze
  - Update split time and maze time to PC
  - Look for centre beam to break. At break, send run time to PC and go to state 5
- 5 Mouse breaking centre gate
  - Waiting to clear centre gate. Send run time and maze time to PC. When clear, go to state 1

Calibration state 0

- At switch on the Arduino sends the values of the phototransistors and trigger points to the PC.
- This is so we can adjust the trigger levels on the potentiometers to account for different lighting conditions.
- Pressing the tactile switch nearest the BBC micro:bit takes us out of the calibration mode in to ready to run state 1 and stores the trigger values
- To calibrate the trigger points for the gates at switch on in state 0
  - Line up lasers with holes in posts with nothing between laser and sensor
  - Turn potentiometer anti clockwise until LED just comes on
  - Block the beam and check that LED goes out
  - Repeat for next sensor

That’s the end of part 1

Any questions?

Hand over to Ian to talk about the interface and what happens at the PC end of things