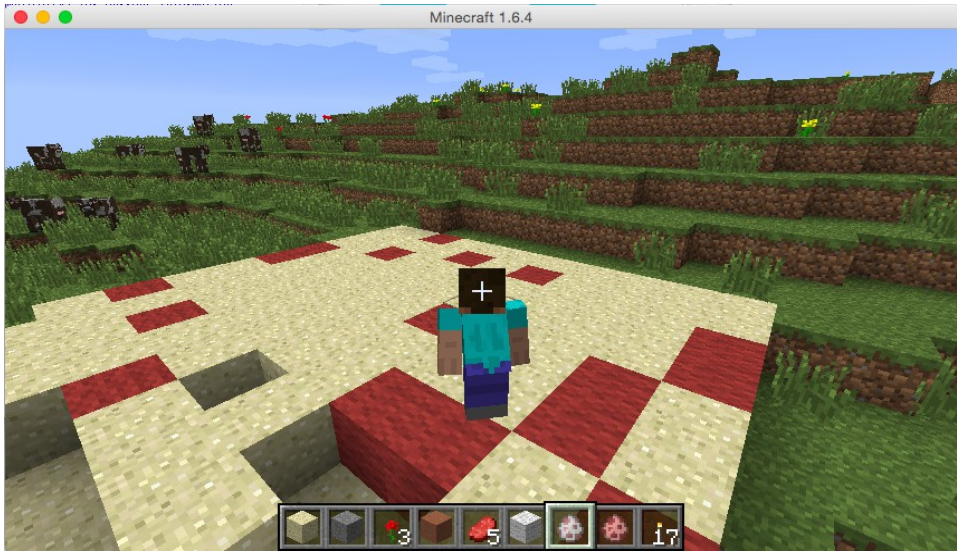
Link Minecraft to electronic circuits, using GPIO



PowerMiner game in use (DEPTH=1)

The Pibrella

shop.pimoroni.com

Using this workbook, you will build an exciting Minecraft game, using electronic circuits to control the game. **This game is called "Power Miner"**. You will write some Python code that links to the running Minecraft game on a Raspberry Pi computer, and you will use more Python code to sense inputs from the real world, and control outputs in the real world, using electronic circuits. To get a link to the real world, you will use a Pibrella plug-on board (but you can use your own buttons and LEDs if you wish).

The object of this game is to find hidden (red) power blocks buried under the ground, by using a very sensitive measuring device. This device is carefully tuned to sense power blocks at a specific depth under the sand. Using your power tester, you have to walk around and test what is under the sand by pressing a button. If you find a power block below your feet, the energy from it is transferred to your power reserves.

But there is a twist! You only have a limited amount of power in your reserves. This energy is used to keep you alive, but is also used to power the tester – use too much power and you will run out of energy! Pit your wits against this game and try to find all the hidden power blocks underground before you run out of energy!

1. Generating hardware output - Flash an LED (LED.py)

```
import time
import RPi.GPIO as GPIO

RED_LED = 27
GPIO.setmode(GPIO.BCM)
GPIO.setup(RED_LED, GPIO.OUT)

while True:
      GPIO.output(RED_LED, True)
      time.sleep(1)
      GPIO.output(RED_LED, False)
      time.sleep(1)
```

At an LXTerminal prompt, type the following to test your program:

```
sudo python LED.py
```

2. Sensing what you are standing on in Minecraft (PowerSensor.py)

```
import time
import RPi.GPIO as GPIO

import mcpi.minecraft as minecraft
import mcpi.block as block
import random

RED_LED = 27
GPIO.setmode(GPIO.BCM)
GPIO.setup(RED_LED, GPIO.OUT)

SIZE = 10
POWER_BLOCKS = 20
DEPTH = 1

mc = minecraft.Minecraft.create()
pos = mc.player.getTilePos()
mc.setBlocks(pos.x, pos.y-1, pos.z, pos.x+SIZE, pos.y-1, pos.z+SIZE, block.SAND.id)

for b in range(POWER_BLOCKS):
      mc.setBlock(pos.x + random.randint(0, SIZE), pos.y-DEPTH, pos.z+random.randint(0,
      SIZE), block.WOOL.id, 14) # RED

while True:
      time.sleep(1)
      pos = mc.player.getTilePos()
      b = mc.getBlock(pos.x, pos.y-DEPTH, pos.z)
      if b == block.WOOL.id:
            GPIO.output(RED_LED, True)
            time.sleep(1)
            GPIO.output(RED_LED, False)
```

The lines in **bold** are added/changed from the previous program.

At an LXTerminal prompt, type the following to test your program:

```
sudo python PowerSensor.py
```

Note: The DEPTH is set to 1, so that you can see the red blocks. This makes it easier to test. Set DEPTH=2 or more to hide the blocks under the sand, to make the game more challenging!

3. Sensing a hardware button to trigger the test (PowerTester.py)

```
import time
import RPi.GPIO as GPIO

import mcpi.minecraft as minecraft
import mcpi.block as block
import random

RED_LED = 27
BUTTON  = 11
GPIO.setmode(GPIO.BCM)
GPIO.setup(RED_LED, GPIO.OUT)
GPIO.setup(BUTTON, GPIO.IN)

SIZE = 10
POWER_BLOCKS = 20
DEPTH = 1

mc = minecraft.Minecraft.create()
pos = mc.player.getTilePos()
mc.setBlocks(pos.x, pos.y-1, pos.z,pos.x+SIZE, pos.y-1, pos.z+SIZE, block.SAND.id)

for b in range(POWER_BLOCKS):
      mc.setBlock(pos.x + random.randint(0, SIZE), pos.y-DEPTH, pos.z+random.randint(0,
      SIZE), block.WOOL.id, 14) # RED

while True:
      time.sleep(1)
      pos = mc.player.getTilePos()
      b = mc.getBlock(pos.x, pos.y-DEPTH, pos.z)
      button = GPIO.input(BUTTON)

      if b == block.WOOL.id and button:
            GPIO.output(RED_LED, True)
            time.sleep(1)
            GPIO.output(RED_LED, False)
```

The lines in **bold** are added/changed from the previous program.

At an LXTerminal prompt, type the following to test your program:

```
sudo python PowerTester.py
```

Note: When you press the button, if the red LED flashes, it means you've found a power pack. If it does not flash, then move around and try again!

4. Generating output inside Minecraft (PowerMinerGame.py)

```
import time
import RPi.GPIO as GPIO

import mcpi.minecraft as minecraft
import mcpi.block as block
import random

RED_LED = 27
AMBER_LED = 17
BUTTON  = 11
GPIO.setmode(GPIO.BCM)
GPIO.setup(RED_LED, GPIO.OUT)
GPIO.setup(AMBER_LED, GPIO.OUT)
GPIO.setup(BUTTON, GPIO.IN)

SIZE = 10
POWER_BLOCKS = 20
DEPTH = 10 # 1 for visible, 2 for hidden, 10 for deep holes!

power = 1000
blocks = POWER_BLOCKS

mc = minecraft.Minecraft.create()
pos = mc.player.getTilePos()
mc.setBlocks(pos.x, pos.y-1, pos.z,pos.x+SIZE, pos.y-1, pos.z+SIZE, block.SAND.id)
mc.setBlocks(pos.x, pos.y, pos.z, pos.x+SIZE, pos.y+SIZE, pos.z+SIZE, block.AIR.id)

for b in range(POWER_BLOCKS):
      mc.setBlock(pos.x + random.randint(0, SIZE), pos.y-DEPTH, pos.z+random.randint(0,
      SIZE), block.WOOL.id, 14) # RED

while power > 0 and blocks > 0:
      time.sleep(1)
      power = power - 1
      mc.postToChat("power:" + str(power) + " blocks:" + str(blocks))

      pos = mc.player.getTilePos()
      b = mc.getBlock(pos.x, pos.y-DEPTH, pos.z)
      button = GPIO.input(BUTTON)
      if button:
            power = power - 10
            if b == block.WOOL.id:
                  GPIO.output(RED_LED, True)
                  time.sleep(1)
                  GPIO.output(RED_LED, False)
                  power = power + 100
                  mc.setBlocks(pos.x, pos.y-DEPTH, pos.z, pos.x, pos.y, pos.z,
                        block.AIR.id)
                  blocks = blocks - 1
            else:
                  GPIO.output(AMBER_LED, True)
                  time.sleep(1)
                  GPIO.output(AMBER_LED, False)

mc.postToChat("game over: final power:" + str(power) + " blocks left:" + str(blocks))
```

The lines in **bold** are added/changed from the previous program.

At an LXTerminal prompt, type the following to test your program:

```
sudo python PowerMinerGame.py
```

Note: The amber LED means you did not find a power block!