Time’s Up! Build a PicoBricks Timer

Picobricks | Beginner | 45 Mins

1 Overview

In this project we are going to create our own countdown timer using PicoBricks.

Measuring time in electronic projects is really important. Think about a washing machine. It needs to know how long to spin the drum in each direction and how long the water needs to flow to dissolve the detergent.

Our timer will work a bit like that. When PicoBricks starts, it shows a welcome message and instructions for using the timer. As you turn the potentiometer, it sets a value. Each press of the button locks that value in and moves you on to the next one: first the hours, then the minutes, then the seconds.

Once the time is set, the countdown begins and the remaining time is shown on the OLED screen. When the timer reaches zero, the buzzer sounds and the LEDs light up until you press the button to stop them.

🎯 What you'll learn

- How to control and count down time within a project
- How to read a potentiometer to set a value
- How to use functions to keep your code tidy and reusable
- How to show numbers and messages on an OLED screen
- How to use a buzzer and LEDs to make an alarm

2 What you'll need

🔧 What you'll need

- PicoBricks Kit
- PicoBricks Brick IDE
- MicroUSB to USB cable
- Laptop or tablet for coding

3 Key words

💡 Key words

Potentiometer

A potentiometer is a knob or dial that controls how much electricity flows through a circuit. Think of it like a tap for water: turn it one way and more comes out, turn it the other way and less comes out.

A volume dial on a speaker is a potentiometer, turn it up for louder sound and down for quieter.

Button

A button is a switch you press to tell your project to do something. The code can check whether it's being pressed or not.

Pressing a doorbell is just like pressing a button, one press sends a signal that makes something happen.

OLED screen

An OLED screen is a small display that shows words, numbers and pictures from your code. On some kits it's labelled the "LED display", but it's the same little screen.

Like the small screen on a fitness watch showing your steps, an OLED screen can show a score or a countdown.

Buzzer

A buzzer is a tiny speaker that makes beeps and tones when you send it electricity. By telling it different numbers, you can make it play notes and tunes.

The beep a microwave makes when your food is ready comes from a buzzer.

Variable

A variable is something that can change. Think of it like a labelled box: you pop something inside, and the label tells you what it is. You can swap what's inside whenever you like.

A variable called score might start at 0, then change to 1, 2 and 3 as you collect points in a game.

Function

A function is a chunk of code you give a name to, so you can use it again and again without building it from scratch each time. Think of it like a favourite recipe, you write down the steps once, then follow it whenever you need it.

Like writing out a pancake recipe once. Whenever you fancy pancakes, you just follow the recipe again rather than working it all out from scratch.

While loop

A while loop keeps repeating its instructions for as long as something stays true. The moment that thing stops being true, the loop stops.

"While it's still raining, keep the umbrella up." As soon as the rain stops, you put the umbrella down, and the loop ends.

Forever loop

A forever loop repeats its instructions again and again without ever stopping (until you switch off or reset). It's perfect for anything you want to keep happening over and over.

Like a clock's second hand that keeps ticking round and round, a forever loop keeps running the same steps over and over.

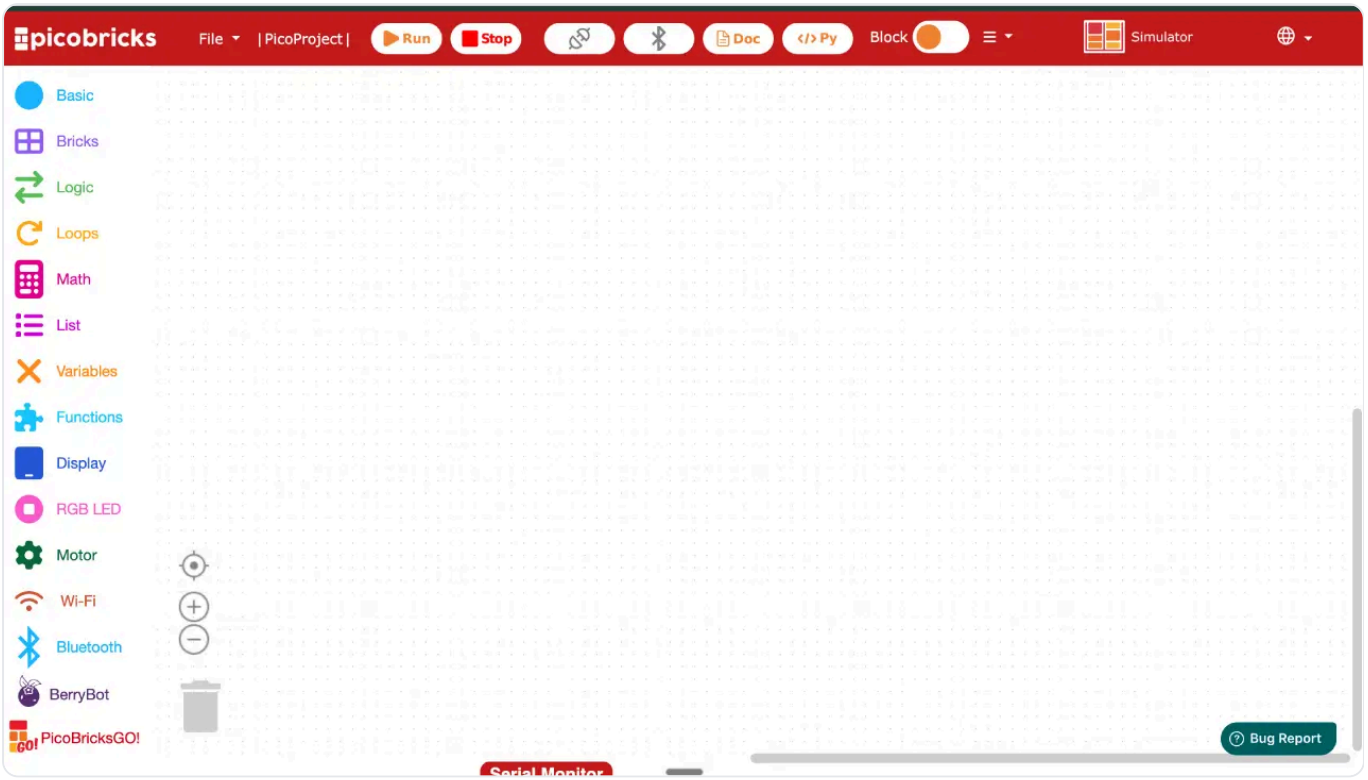
If statement

An if statement checks whether something is true, and only runs its instructions when it is. It's how code makes decisions.

"If it's cold, put on a coat." If it's warm, you skip the coat, the action only happens when the check is true.

4 Set up your PicoBricks

1. Open your favourite web browser. We recommend Google Chrome or Microsoft Edge.
2. Type ide.picobricks.com into the address bar to open the PicoBricks coding editor.
3. Connect the PicoBricks board to your computer with the USB cable.
4. Select the **Connection** button at the top of your code editor to pair the Raspberry Pi Pico with the code editor.



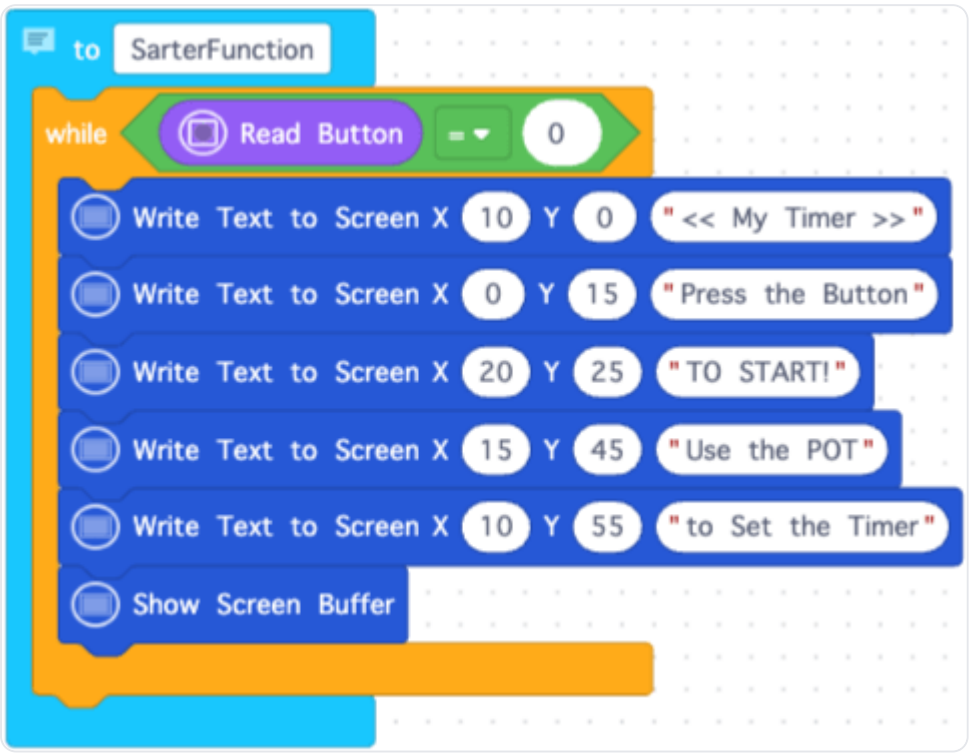
5 Build the starter function

First, we will create a function that shows a welcome message on the screen. This tells the person using the timer what to do before they start.

1. Select **Functions**. Select a `to do something` block and drop it within the code area.
2. Change **do something** to **StarterFunction**.
3. Select **Loops**. Select a while block and snap it inside the `to StarterFunction` block.
4. Select **Logic**. Select an `=` block and snap it onto the `while` block.
5. Select **Bricks**. Select a `Read Button` block and snap it onto the `while` block before the **=** sign.
6. Select **Math**. Select a `0` block and snap it onto the `while` block after the **=** sign.
7. Select **Display**. Select a `Write Text to Screen` block and snap it inside the `while Read Button = 0` block.
8. Change the **X** value to **10** and change **PicoBricks** to **<< My Timer >>**.
9. Select **Display**. Select another `Write Text to Screen` block and snap it under the first one.
10. Change the **Y** value to **15** and change **PicoBricks** to **Press the Button**.
11. Select **Display**. Select another `Write Text to Screen` block and snap it under that.
12. Change **PicoBricks** to **To START!**
13. Select **Display**. Select another `Write Text to Screen` block and snap it underneath.
14. Change the **X** value to **15**, the **Y** value to **45**, and change **PicoBricks** to **Use the POT**.

15. Select **Display**. Select another `Write Text to Screen` block and snap it underneath.
16. Change the **X** value to **10**, the **Y** value to **55**, and change **PicoBricks** to **to Set the Timer**.
17. Select **Display**. Select a `Show Screen Buffer` block and snap it underneath.

> **POT** is short for potentiometer. That is the little dial we will turn to set the timer.



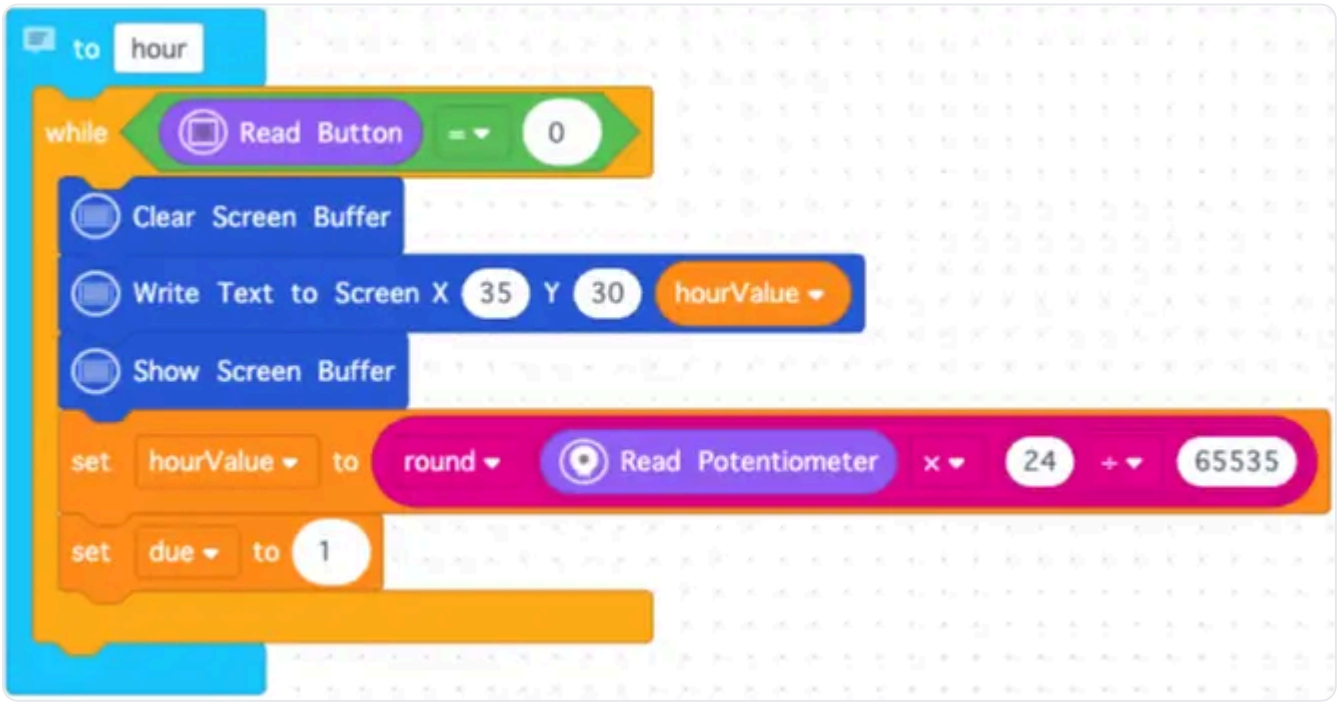
6 Build the hour function

Next, we will create the hour function. This lets us use the potentiometer to choose how many hours we want on the timer.

1. Select **Functions**. Select a `to do something` block and drop it within the code area.
2. Change **do something** to **hour**.
3. Select **Loops**. Select a `while` block and snap it inside the `to hour` block.
4. Select **Logic**. Select an `=` block and snap it onto the `while` block.
5. Select **Bricks**. Select a `Read Button` block and snap it onto the `while` block before the **=** sign.
6. Select **Math**. Select a `0` block and snap it onto the `while` block after the **=** sign.
7. Select **Display**. Select a `Clear Screen Buffer` block and snap it inside the `while Read Button = 0` block.
8. Select **Display**. Select a `Write Text to Screen` block and snap it under the `Clear Screen Buffer` block.
9. Change the **X** value to **35** and the **Y** value to **30**.

10. Select **Variables**. Select **Create Variable** and type **hourValue**.
11. Select a `hourValue` block and snap it onto the `Write Text to Screen` block where it says **PicoBricks**.
12. Select **Display**. Select a `Show Screen Buffer` block and snap it underneath.
13. Select **Variables**. Select a `set hourValue to` block and snap it under the `Show Screen Buffer` block.
14. Select **Math**. Select a `round` block and snap it onto the `set hourValue to` block.
15. Select **Math**. Select a `1 + 1` block and snap it onto the `round` block where it says **3.1**.
16. Change the **+** to **x**.
17. Select **Bricks**. Select a `Read Potentiometer` block and snap it into the first **1** of the maths block.
18. Select **Math**. Select a `1 + 1` block and snap it into the second **1** of the maths block.
19. Change the first **1** to **24**, the **+** sign to **÷**, and the remaining **1** to **65535**.
20. Select **Variables**. Select **Create Variable** and type **due**.
21. Select a `set due to` block and snap it below the `set hourValue to round Read Potentiometer x 24 ÷ 65535` block.
22. Select **Math**. Select a `0` block and snap it onto the `set due to` block.
23. Change the **0** to **1**.

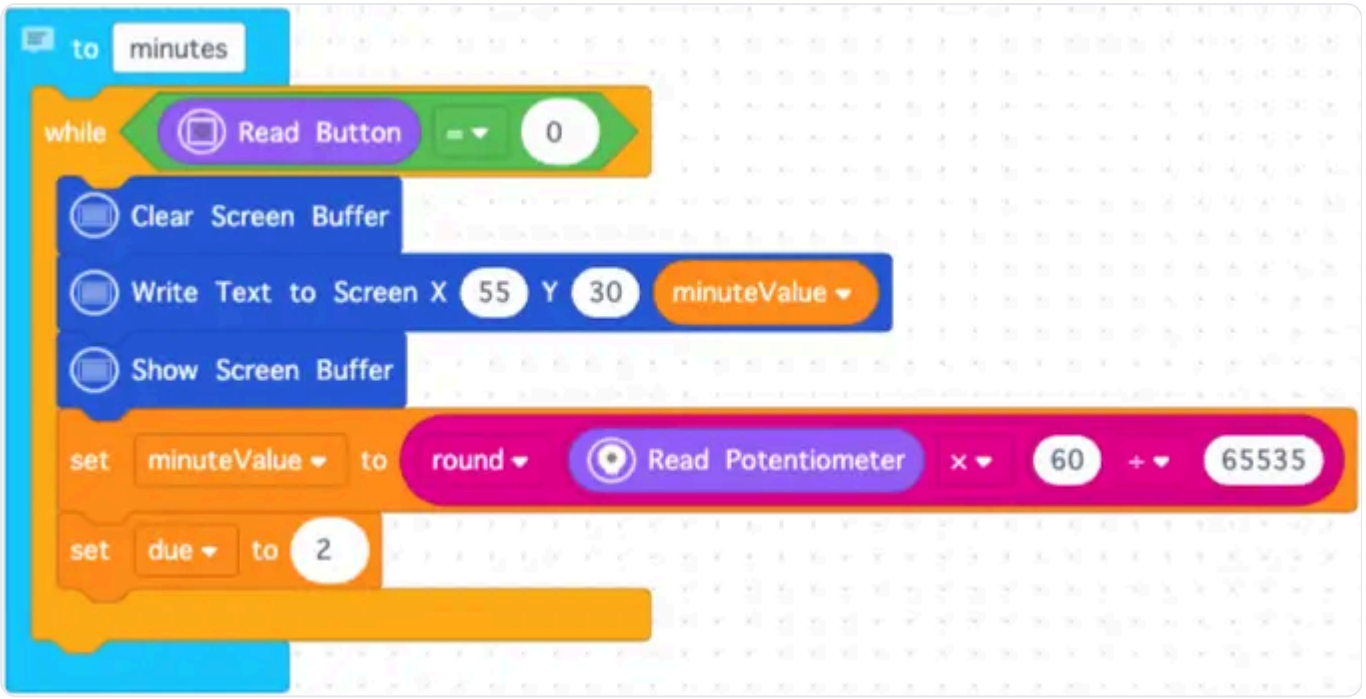
🧠 This part turns the dial reading into a number of hours. The `due` variable helps the code know which part of the timer setup should happen next.



7 Build the minutes function

The minutes function works almost the same way as the hour function, but this time it sets the minutes.

1. Select **Functions**. Select a `to do something` block and drop it in the code area.
2. Change **do something** to **minutes**.
3. Select **Loops**. Select a `while` block and snap it inside the `to minutes` block.
4. Select **Logic**. Select an `=` block and snap it onto the `while` block.
5. Select **Bricks**. Select a `Read Button` block and snap it onto the `while` block before the **=** sign.
6. Select **Math**. Select a `0` block and snap it onto the `while` block after the **=** sign.
7. Select **Display**. Select a `Clear Screen Buffer` block and snap it within the `while Read Button = 0` block.
8. Select **Display**. Select a `Write Text to Screen` block and snap it under the `Clear Screen Buffer` block.
9. Change the **X** value to **55** and the **Y** value to **30**.
10. Select **Variables**. Select **Create Variable** and type **minuteValue**.
11. Select a `minuteValue` block and snap it onto the `Write Text to Screen` block where it says **PicoBricks**.
12. Select **Display**. Select a `Show Screen Buffer` block and snap it underneath.
13. Select **Variables**. Select a `set minuteValue to` block and snap it under the `Show Screen Buffer` block.
14. Select **Math**. Select a `round` block and attach it within the `set minuteValue to` block.
15. Select **Math**. Select a `1 + 1` block and attach it within the **3.1** of the `round` block.
16. Change the **+** sign to **x**.
17. Select **Bricks**. Select a `Read Potentiometer` block and snap it within the first **1** of the maths block.
18. Select **Math**. Select a `1 + 1` block and snap it within the second **1** of the maths block.
19. Change the first **1** to **60**, the **+** to **÷**, and the remaining **1** to **65535**.
20. Select **Variables**. Select a `set minuteValue to` block and snap it underneath.
21. Change **minuteValue** to **due**.
22. Select **Math**. Select a `0` block and snap it onto the `set due to` block.
23. Change **0** to **2**.

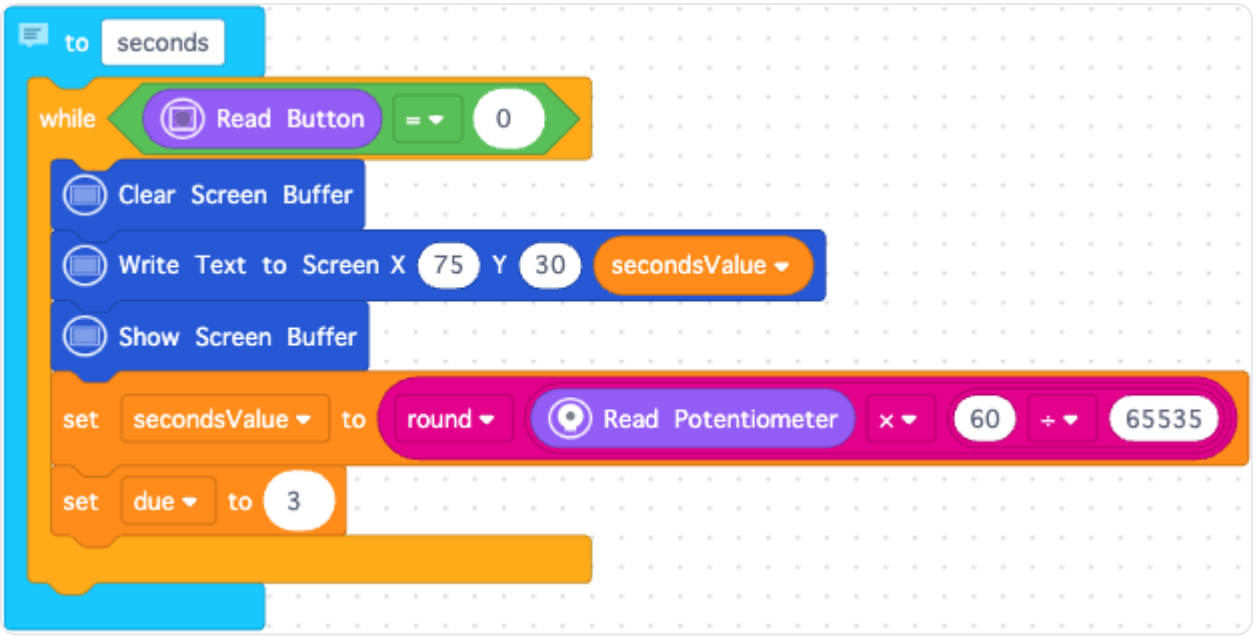


8 Build the seconds function

Now we will create the seconds function. This sets the seconds before the countdown begins.

1. Select **Functions**. Select a `to do something` block and drop it in the code area.
2. Change **do something** to **seconds**.
3. Select **Loops**. Select a `while` block and snap it within the `to seconds` block.
4. Select **Logic**. Select an `=` block and snap it onto the `while` block.
5. Select **Bricks**. Select a `Read Button` block and snap it onto the `while` block before the **=**.
6. Select **Math**. Select a `0` block and snap it onto the `while` block after the **=**.
7. Select **Display**. Select a `Clear Screen Buffer` block and snap it within the `while Read Button = 0` block.
8. Select **Display**. Select a `Write Text to Screen` block and snap it under the `Clear Screen Buffer` block.
9. Change the **X** value to **75** and the **Y** value to **30**.
10. Select **Variables**. Select **Create Variable** and type **secondsValue**.
11. Select a `secondsValue` block and snap it onto the `Write Text to Screen` block where it says **PicoBricks**.
12. Select **Display**. Select a `Show Screen Buffer` block and snap it underneath.
13. Select **Variables**. Select a `set secondsValue to` block and snap it under the `Show Screen Buffer` block.
14. Select **Math**. Select a `round` block and snap it onto the `set secondsValue to` block.
15. Select **Math**. Select a `1 + 1` block and snap it onto the `round` block where it says **3.1**.
16. Select **Bricks**. Select a `Read Potentiometer` block and snap it into the first **1**.

17. Change the **+** sign to **x**.
18. Select **Math**. Select a `1 + 1` block and snap it into the second **1**.
19. Change the first **1** to **60**, the **+** to **÷**, and the remaining **1** to **65535**.
20. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
21. Change **secondsValue** to **due**.
22. Select **Math**. Select a `0` block and snap it to the `set due to` block.
23. Change the **0** to **3**.

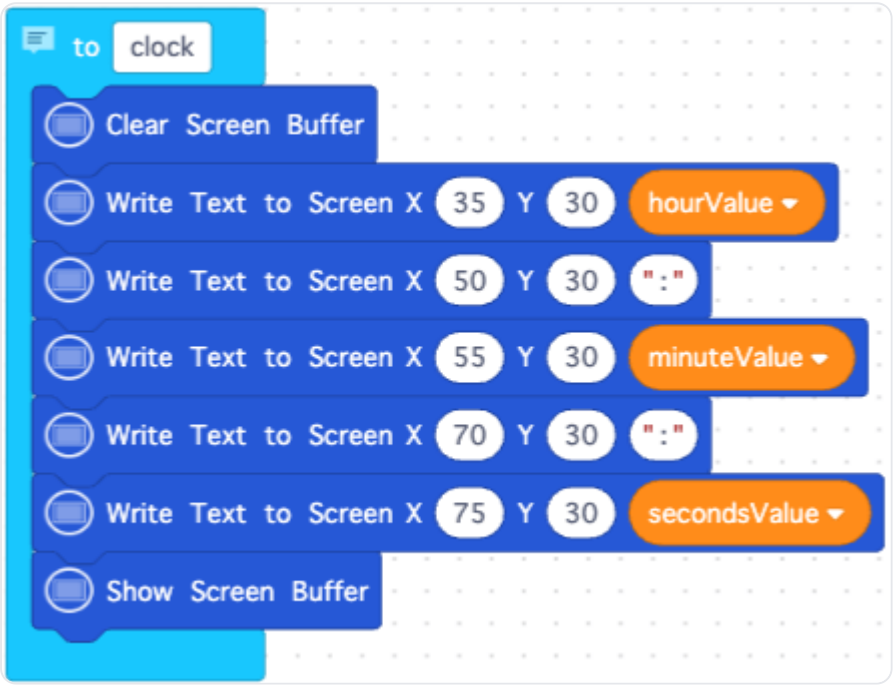


9 Build the clock function

The clock function puts the hours, minutes and seconds together on the OLED screen.

1. Select **Functions**. Select a `to do something` block and drop it in the code area.
2. Change **do something** to **clock**.
3. Select **Display**. Select a `Clear Screen Buffer` block and snap it within the `to clock` block.
4. Select **Display**. Select a `Write Text to Screen` block and snap it under the `Clear Screen Buffer` block.
5. Change the **X** value to **35** and the **Y** value to **30**.
6. Select **Variables**. Select a `hourValue` block and snap it onto the `Write Text to Screen` block where it says **PicoBricks**.
7. Select **Display**. Select a `Write Text to Screen` block and snap it underneath.
8. Change the **X** value to **50**, the **Y** value to **30**, and change **PicoBricks** to **:**.
9. Select **Display**. Select another `Write Text to Screen` block and snap it underneath.
10. Change the **X** value to **55** and the **Y** value to **30**.

11. Select **Variables**. Select a `minuteValue` block and snap it onto the block where it says **PicoBricks**.
12. Select **Display**. Select another `Write Text to Screen` block and snap it underneath.
13. Change the **X** value to **70**, the **Y** value to **30**, and change **PicoBricks** to **:**.
14. Select **Display**. Select another `Write Text to Screen` block and snap it underneath.
15. Change the **X** value to **75** and the **Y** value to **30**.
16. Select **Variables**. Select a `secondsValue` block and snap it onto the block where it says **PicoBricks**.
17. Select **Display**. Select a `Show Screen Buffer` block and snap it underneath.



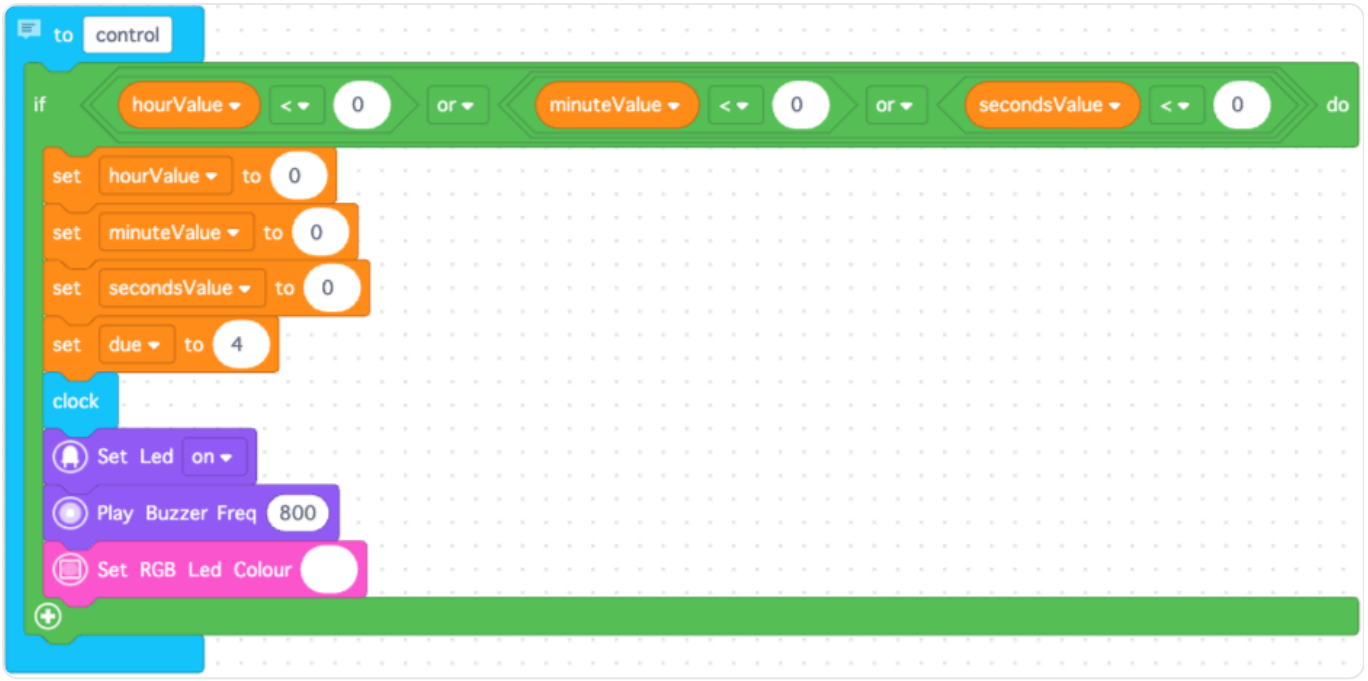
10 Build the control function

The control function checks whether the countdown has finished. When the timer reaches zero, it switches on the alarm.

1. Select **Functions**. Select a `to do something` block and drop it in the code area.
2. Change **do something** to **control**.
3. Select **Logic**. Select an if block and snap it within the `to control` block.
4. Select **Logic**. Select an `and` block and snap it onto the `if` block.
5. Select **Logic**. Select an `=` block and snap it within the `if` block before the **and**.
6. Select **Variables**. Select a `hourValue` block and snap it onto the `if` block before the **=** sign.
7. Change the **=** sign to a **<** sign.
8. Select **Math**. Select a `0` block and snap it onto the `if` block before the **and**.
9. Change the **and** to **or**.

10. Select **Logic**. Select an `and` block and snap it onto the `if` block after the **or**.
11. Select **Logic**. Select an `=` block and snap it onto the `if` block before the **and**.
12. Select **Variables**. Select a `minuteValue` block and snap it onto the `if` block within the blank space after the **or**.
13. Change the **=** to **<**.
14. Select **Math**. Select a `0` block and snap it onto the `if` block after the **<** symbol.
15. Change the **and** to **or**.
16. Select **Logic**. Select an `=` block and snap it onto the `if` block after the **or**.
17. Select **Variables**. Select a `secondsValue` block and snap it onto the `if` block after the **or**.
18. Change the **=** to a **<** symbol.
19. Select **Math**. Select a `0` block and snap it within the remaining blank space of the `if` block.
20. Select **Variables**. Select a `set secondsValue to` block and snap it within the `if hourValue < 0 or minuteValue < 0 or secondsValue < 0 do` block.
21. Change **secondsValue** to **hourValue**.
22. Select **Math**. Select a `0` block and snap it onto the `set hourValue to` block.
23. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
24. Change **secondsValue** to **minuteValue**.
25. Select **Math**. Select a `0` block and snap it onto the `set minuteValue to` block.
26. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
27. Select **Math**. Select a `0` block and snap it onto the `set secondsValue to` block.
28. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
29. Change **secondsValue** to **due**.
30. Select **Math**. Select a `0` block and snap it onto the `set due to` block.
31. Change **0** to **4**.
32. Select **Functions**. Select a `clock` block and snap it underneath.
33. Select **Bricks**. Select a `Set Led` block and snap it underneath.
34. Select **Bricks**. Select a `Play Buzzer Freq` block and snap it underneath.
35. Change **300** to **800**.
36. Select **RGB LED**. Select a `Set RGB Led Colour` block and snap it underneath.

This is the alarm part of the timer. When the countdown finishes, the buzzer sounds and the lights turn on.



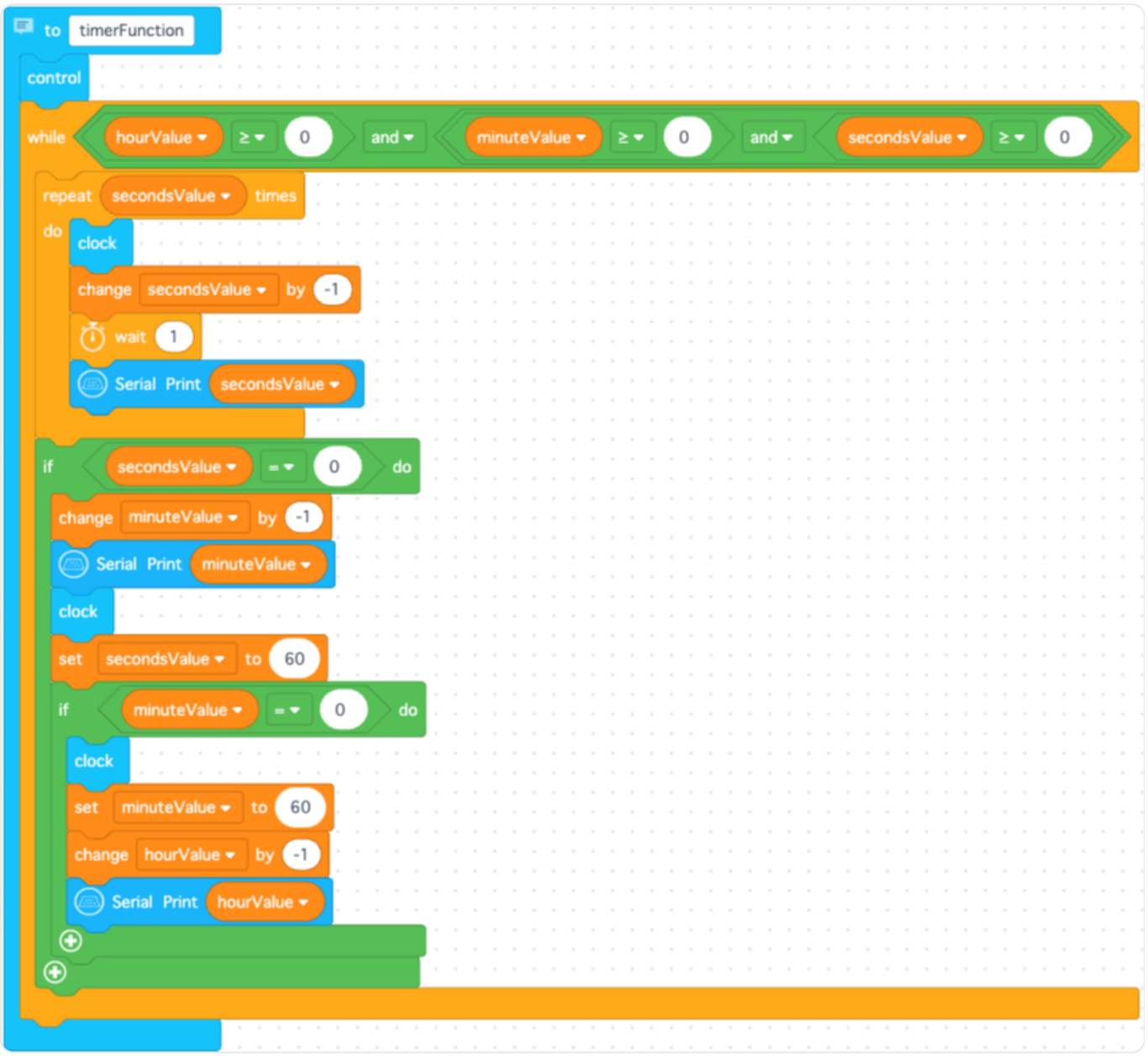
11 Build the timer function

Now it is time to build the timer function. This is the part that counts down the time.

1. Select **Functions**. Select a `to do something` block and drop it in the code area.
2. Change **do something** to **timerFunction**.
3. Select **Functions**. Select a `control` block and snap it within the `timerFunction` block.
4. Select **Loops**. Select a `while` block and snap it underneath the `control` block.
5. Select **Logic**. Select an `and` block and snap it onto the `while` block.
6. Select **Logic**. Select an `=` block and snap it onto the `while` block before the **and**.
7. Select **Variables**. Select a `hourValue` block and snap it onto the `while` block before the **=** sign.
8. Change the **=** sign to a **≥** sign.
9. Select **Math**. Select a `0` block and snap it after the **≥** sign.
10. Select **Logic**. Select an `and` block and snap it onto the `while` block after the **and**.
11. Select **Logic**. Select an `=` block and snap it onto the `while` block between the two **and** blocks.
12. Select **Variables**. Select a `minuteValue` block and snap it onto the `while` block before the **=** sign.
13. Change the **=** sign to a **≥** sign.
14. Select **Math**. Select a `0` block and snap it after the **≥** sign.
15. Select **Logic**. Select an `=` block and snap it onto the `while` block after the **and**.
16. Select **Variables**. Select a `secondsValue` block and snap it onto the `while` block before the **=** sign.

17. Change the **=** sign to a **≥** sign.
18. Select **Math**. Select a `0` block and snap it after the **≥** sign.
19. Select **Loops**. Select a `repeat` block and snap it within the `while hourValue ≥ 0 and minuteValue ≥ 0 and secondsValue ≥ 0` block.
20. Select **Variables**. Select a `secondsValue` block and snap it onto the `repeat` block.
21. Select **Functions**. Select a `clock` block and snap it within the `repeat secondsValue times` block.
22. Select **Variables**. Select a `change secondsValue by` block and snap it under the `clock` block.
23. Change the **1** to **-1**.
24. Select **Loops**. Select a `wait` block and snap it underneath.
25. Select **Basic**. Select a `Serial Print` block and snap it underneath.
26. Select **Variables**. Select a `secondsValue` block and snap it onto the `Serial Print` block where it says **PicoBricks**.
27. Select **Logic**. Select an `if` block and snap it under the `repeat secondsValue times` block.
28. Select **Logic**. Select an `=` block and snap it onto the `if` block.
29. Select **Math**. Select a `0` block and snap it onto the `if` block before the **=** sign.
30. Select **Variables**. Select a `change secondsValue by` block and snap it within the `if secondsValue = 0` block.
31. Change **secondsValue** to **minuteValue** and **1** to **-1**.
32. Select **Basic**. Select a `Serial Print` block and snap it underneath.
33. Select **Variables**. Select a `minuteValue` block and snap it onto the `Serial Print` block where it says **PicoBricks**.
34. Select **Functions**. Select a `clock` block and snap it underneath.
35. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
36. Change **1** to **60**.
37. Select **Logic**. Select an `if` block and snap it underneath.
38. Select **Logic**. Select an `=` block and snap it onto the `if` block.
39. Select **Variables**. Select a `minuteValue` block and snap it onto the `if` block before the **=** sign.
40. Select **Math**. Select a `0` block and snap it onto the `if` block after the **=** sign.
41. Select **Functions**. Select a `clock` block and snap it within the `if minuteValue = 0` block.
42. Select **Variables**. Select a `set secondsValue to` block and snap it under the `clock` block.
43. Change **secondsValue** to **minuteValue**.

14. Select **Math**. Select a `0` block and snap it onto the `set minuteValue to` block.
15. Change the **0** to **60**.
16. Select **Variables**. Select a `change secondsValue by` block and snap it underneath.
17. Change **secondsValue** to **hourValue** and **1** to **-1**.
18. Select **Basic**. Select a `Serial Print` block and snap it underneath.
19. Select **Variables**. Select a `hourValue` block and snap it onto the `Serial Print` block where it says **PicoBricks**.



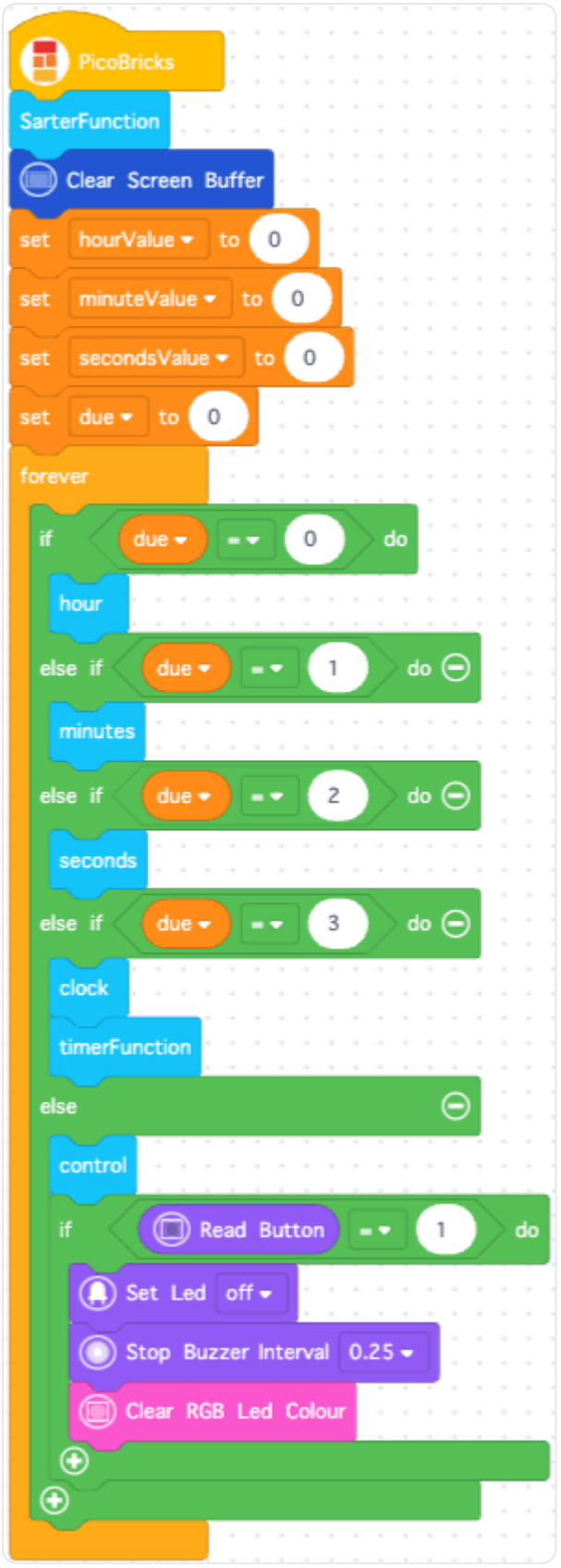
12 Build the starter code

We have now created all the functions. The last job is to create the main code that runs when PicoBricks starts.

1. Select **Basic**. Select a `PicoBricks` block and drop it in the code area.

2. Select **Functions**. Select a `StarterFunction` block and snap it to the `PicoBricks` block.
3. Select **Display**. Select a `Clear Screen Buffer` block and snap it underneath.
4. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
5. Change **secondsValue** to **hourValue**.
6. Select **Math**. Select a `0` block and snap it onto the `set hourValue to` block.
7. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
8. Change **secondsValue** to **minuteValue**.
9. Select **Math**. Select a `0` block and snap it onto the `set minuteValue to` block.
10. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
11. Select **Math**. Select a `0` block and snap it onto the `set secondsValue to` block.
12. Select **Variables**. Select a `set secondsValue to` block and snap it underneath.
13. Change **secondsValue** to **due**.
14. Select **Math**. Select a `0` block and snap it onto the `set due to` block.
15. Select **Loops**. Select a forever block and snap it underneath.
16. Select **Logic**. Select an `if` block and snap it within the `forever` block.
17. Select **Logic**. Select an `=` block and snap it onto the `if` block.
18. Select **Variables**. Select a `due` block and snap it onto the `if` block before the **=** sign.
19. Select **Math**. Select a `0` block and snap it onto the `if` block after the **=** sign.
20. Select **Functions**. Select a `hour` block and snap it into the `if due = 0 do` block.
21. Select the **+** on the `if` block two times to create an `else if` block.
22. Select **Logic**. Select an `=` block and snap it onto the `else if` block.
23. Select **Variables**. Select a `due` block and snap it onto the `else if` block before the **=** sign.
24. Select **Math**. Select a `0` block and snap it onto the `else if` block after the **=** sign.
25. Change the **0** to **1**.
26. Select **Functions**. Select a `minutes` block and snap it within the `else if due = 1 do` block.
27. Select the **+** sign at the bottom of the `if` block to create another `else if` block.
28. Select **Logic**. Select an `=` block and snap it onto the `else if` block.
29. Select **Variables**. Select a `due` block and snap it onto the `else if` block before the **=** sign.
30. Select **Math**. Select a `0` block and snap it onto the `else if` block after the **=** sign.
31. Change the **0** to **2**.
32. Select **Functions**. Select a `seconds` block and snap it within the `else if due = 2 do` block.
33. Select the **+** sign at the bottom of the `if` block to create another `else if` block.

34. Select **Logic**. Select an `=` block and snap it onto the `else if` block.
35. Select **Variables**. Select a `due` block and snap it onto the `else if` block before the **=** sign.
36. Select **Math**. Select a `0` block and snap it onto the `else if` block after the **=** sign.
37. Change the **0** to **3**.
38. Select **Functions**. Select a `clock` block and snap it within the `else if due = 3 do` block.
39. Select **Functions**. Select a `timerFunction` block and snap it under the `clock` block.
40. Select **Functions**. Select a `control` block and snap it within the `else` block.
41. Select **Logic**. Select an `if` block and snap it under the `control` block.
42. Select **Logic**. Select an `=` block and snap it onto the `if` block.
43. Select **Bricks**. Select a `Read Button` block and snap it onto the `if` block before the **=** sign.
44. Select **Math**. Select a `0` block and snap it onto the `if` block after the **=** sign.
45. Change **0** to **1**.
46. Select **Bricks**. Select a `Set Led` block and snap it within the `if Read Button = 1 do` block.
47. Change **on** to **off**.
48. Select **Bricks**. Select a `Stop Buzzer Interval` block and snap it underneath.
49. Select **RGB LED**. Select a `Clear RGB Led Colour` block and snap it underneath.



13 Run it and test

We have now completed all the code that we need. The next thing to do is test it and make sure it works as expected.

1. Hit the **Run** button at the top of the code area to start the PicoBricks.
2. You will see the startup message on the OLED display telling you to press the button to start.
3. Press the button to start setting the timer.
4. Turn the potentiometer to choose the number of hours.
5. Press the button to lock in the hours and move to minutes.
6. Turn the potentiometer to choose the number of minutes.
7. Press the button to lock in the minutes and move to seconds.
8. Turn the potentiometer to choose the number of seconds.
9. Press the button to lock in the seconds and start the countdown.
10. When the timer reaches zero, the buzzer sounds and the LED and RGB LED light up.
11. Press the button to stop the alarm.

⏰ You have built your own countdown timer with PicoBricks. Brilliant work!

14 Try it yourself

🚀 Try it yourself

Challenge: Can you make the alarm more exciting? Try changing the buzzer frequency so it plays a little tune when the timer reaches zero, or make the RGB LED flash through different colours instead of staying on one.

For an extra challenge, add a message on the screen that says **Time's up!** when the countdown finishes.

15 Stuck? Quick fixes

Problem	Try this
Nothing appears on the OLED screen	Check the screen module is connected properly, and make sure you've added a `Show Screen Buffer` block after writing your text. Without it, the screen stays blank.
The timer won't start when I press the button	Make sure the button module is connected. Remember you need to press the button to set the hours, minutes and seconds before the countdown begins.
The numbers don't change when I turn the dial	Check the potentiometer is connected, and double-check you used a `Read Potentiometer` block in your hour, minutes and seconds functions.
My code won't upload to the PicoBricks	Check the USB cable is plugged in, then press the connection button at the top of the editor to pair the Raspberry Pi Pico again.
The buzzer doesn't sound at zero	Make sure the buzzer module is connected, and check your `control` function includes the `Play Buzzer Freq` block.