KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Analog Sensors - Reflectance

Professional Development Workshop

Analog Sensors - Reflectance

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Analog Sensors - Reflectance

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Slide	Topic
13	Plug in the Light Sensor
14	Starting with a Light
15	wait_for_light Calibration Routine
16-17	Starting with a Light
18	Starting Programs with a Light
19	wait_for_light Function
20	Plug in the Light Sensor
21	Starting with a Light
22	wait_for_light Calibration Routine

Analog Sensor:
Small Top Hat Sensors

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

This is a reflectance sensor that works at short distances. There is an infrared (IR) emitter and an IR collector in this sensor. The IR emitter sends out IR light and the IR collector measures how much is reflected back.

Amount of IR reflected back depends on surface texture, color and distance to surface among other factors. This sensor is excellent for line following

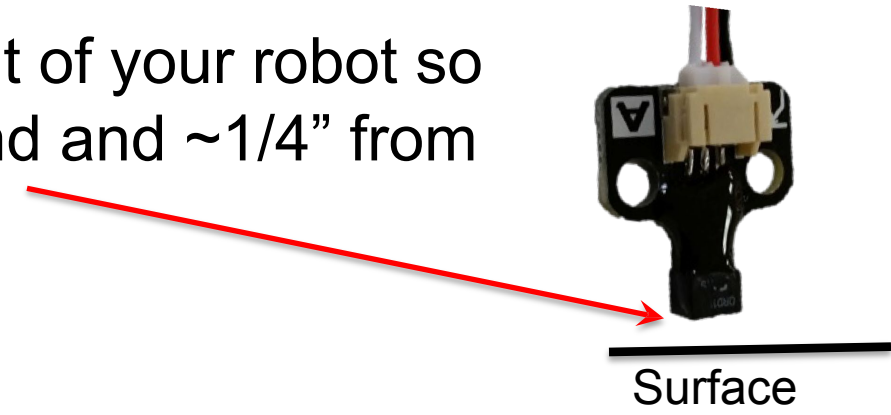
Black materials typically absorb IR and reflect very little IR while white materials typically absorb little IR and reflect most of it back

- If this sensor is mounted at a fixed height above a surface, it is easy to distinguish a dark color from a light color
- Connect to an analog port (0 to 5)

Reflectance Sensor Ports

This is an **analog** sensor, so plug it into any of your analog ports 0 through 5

- Values returned can be between 0 and 4095
- Mount the sensor on the front of your robot so that it is pointing to the ground and ~1/4” from the surface



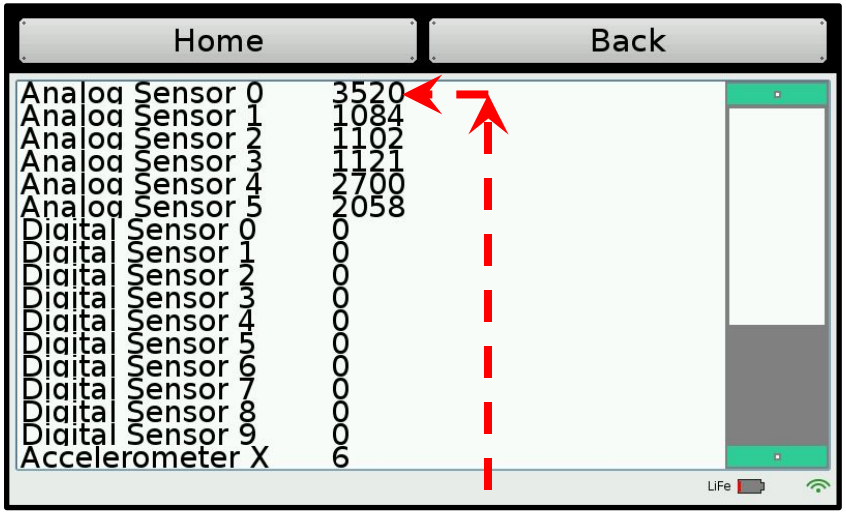
Reading Sensor Values From the Sensor List

KISS INSTITUTE FOR PRACTICAL ROBOTICS

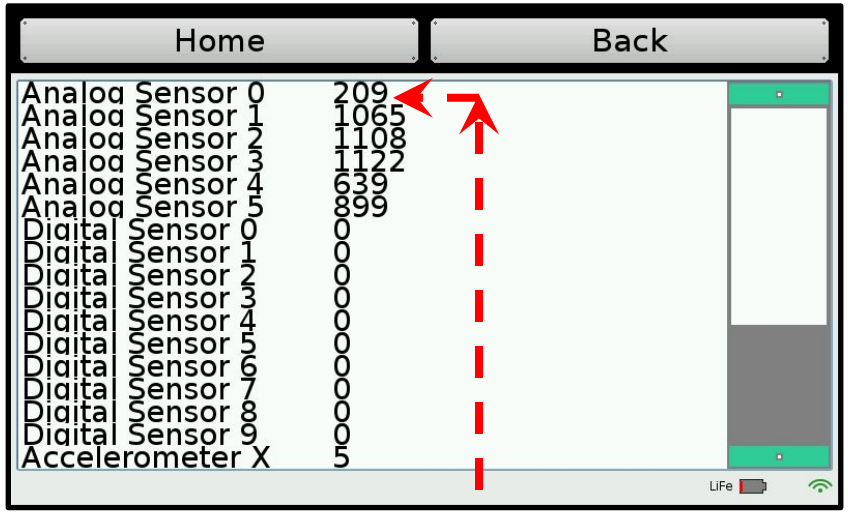
Botball®

With the IR sensor plugged into analog port #0

- Over a white surface the value is (~200)
- Over a black surface the value is (~3200)



Your IR sensor is correctly mounted when you have values between ~2900 and ~3800 on the Black Surface



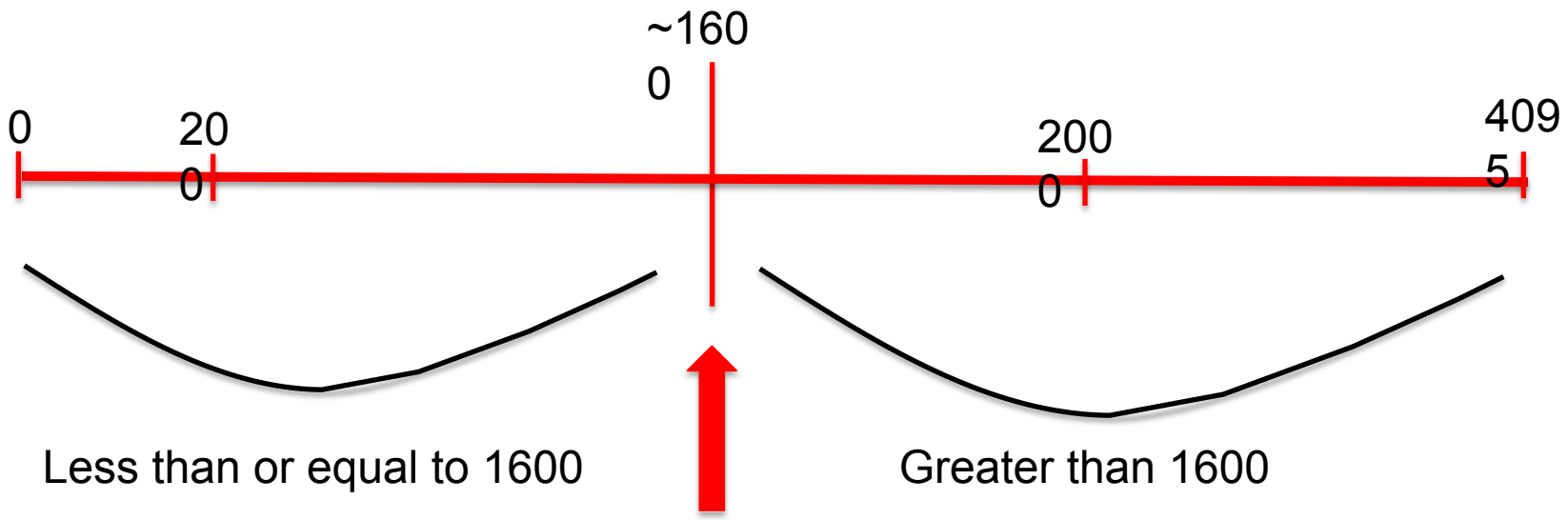
Your IR sensor is correctly mounted when you have values between ~175 and ~300 on the White Surface.

Understanding the IR Values

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

- Place your IR analog sensor in one of the analog ports (0 to 5).
- After mounting your IR sensor, check value when sensor is over black on Mat A, B or black tape



My black ***threshold*** value is ~1600

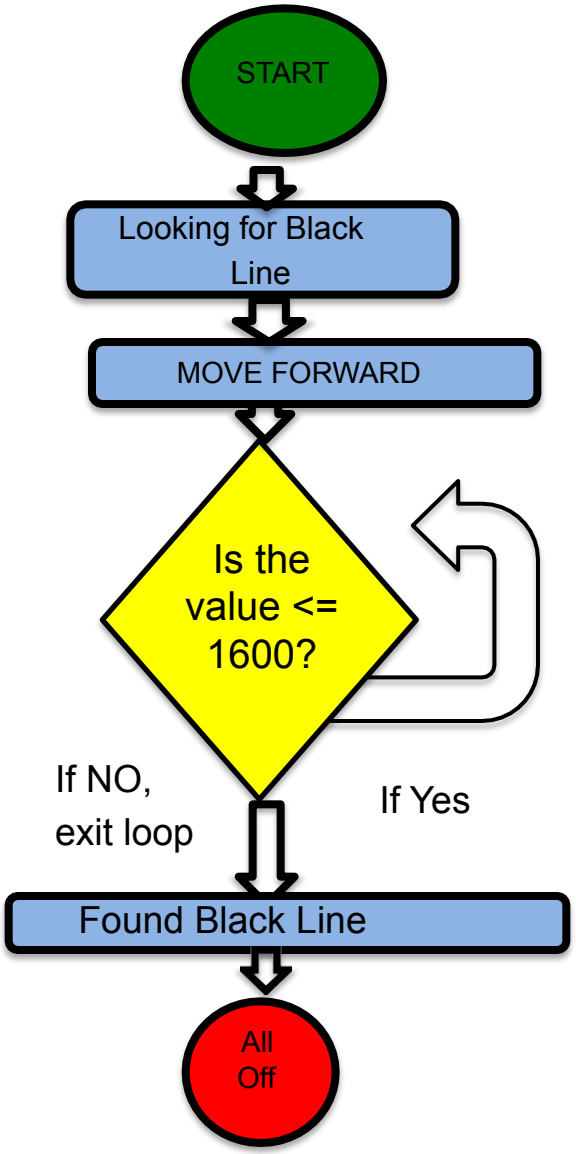
Find the Black Line

KISS INSTITUTE FOR PRACTICAL ROBOTICS

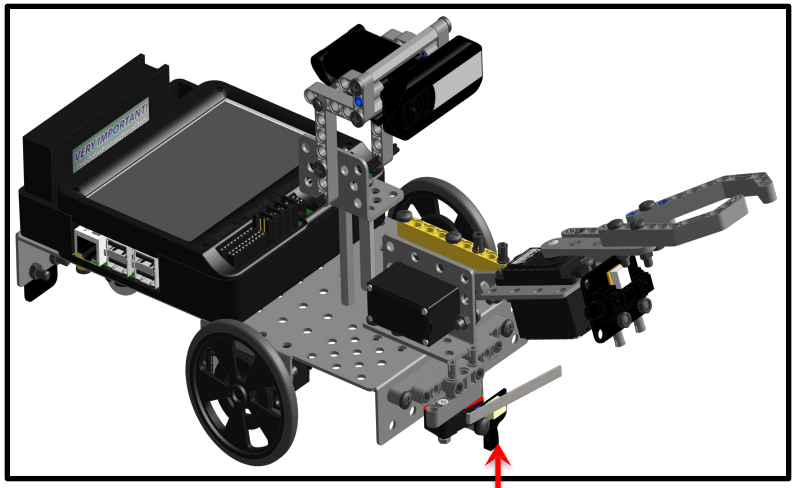
Botball®

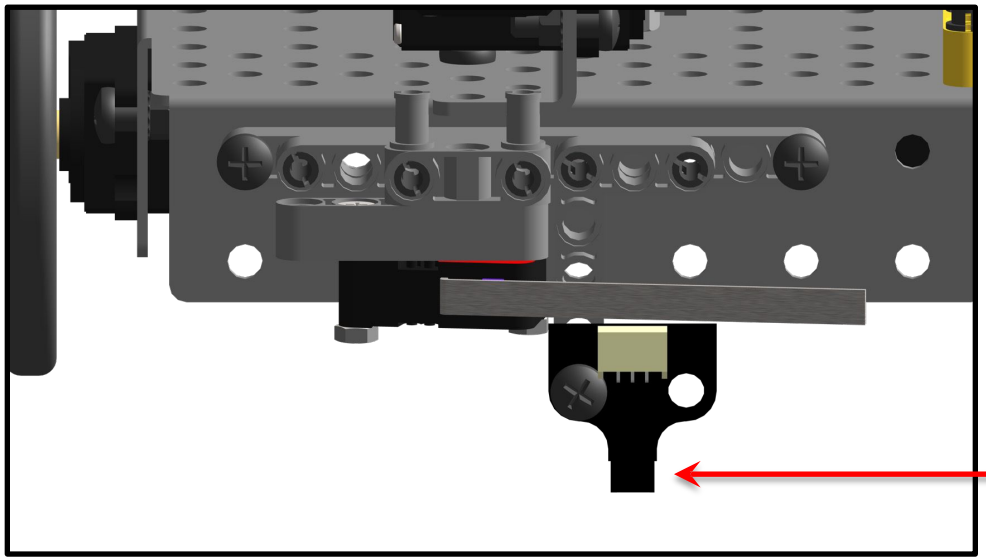
Pseudocode (Task Analysis)

1. Prints looking for black line
2. Check the sensor value in analog port 0, <= 1600
3. Drive forward as long as the value is <= 1600
4. Exit loop when value is 1600 or greater
5. Shut everything off



Mounted Sensor on DemoBot





while “find black line” Solution

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Source Code

```
1  #include <kipr/wombat.h>
2
3  int main()
4  {
5      printf(“Find the black line\n”);
6      while (analog(0) < 1600)
7      {
8          motor(0, 78);
9          motor(3, 74); // why slightly less?
10     }
11     ao();
12
13     return 0;
14 }
15
```

Starting Programs with a Light

The light sensor is a cool way to automatically start your robot and critical for Botball robots at the beginning of the game.

The **wait_for_light ( )** function allows your program to run after your robot senses a light.

Note: It has a built-in calibration routine that will come up on the screen (a step-by-step guide for this calibration routine is on a following slide).

The light sensor senses infrared light, so light must be emitted from an incandescent light, not an LED light.

For our activities, you can use a flashlight.

The more light (infrared) detected, the lower the reported value.

wait_for_light Function

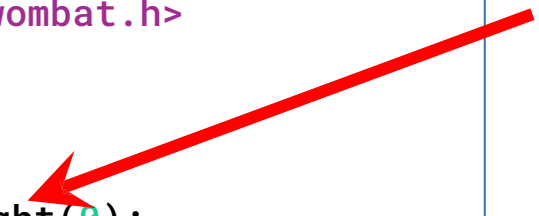
KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

```
wait_for_light(0);
// Waits for the light on port #0 before going to the next line.
```

Source Code

```
1  #include <kipr/wombat.h>
2
3  int main()
4  {
5      wait_for_light(0);
6      printf("I see the light!\n");
7      return 0;
8  }
9
10
```

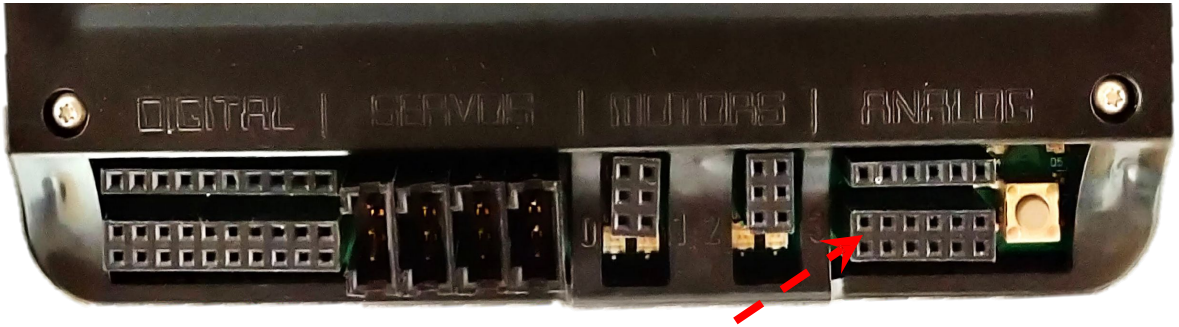


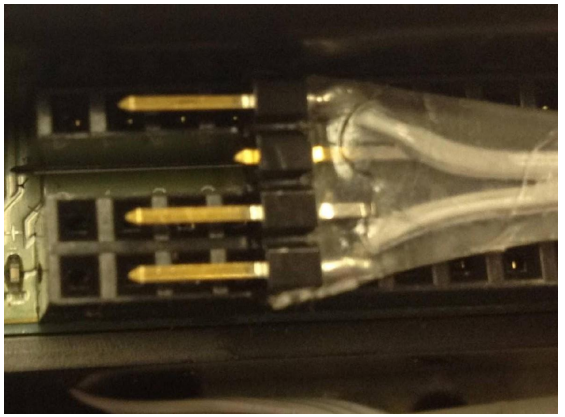
What is this?

Plug in the Light Sensor

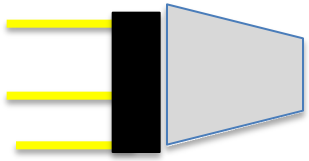
KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®





Sensor plug orientation



Plug your light sensor into analog port 0

Starting with a Light

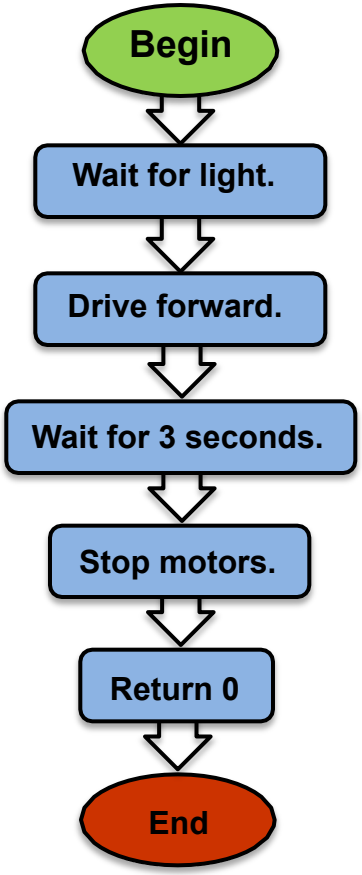
KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Description: Write a program that waits for a light to come on, drives the DemoBot forward for 3 seconds, and then stops.

Analysis: What is the program supposed to do?

Flowchart



Pseudocode

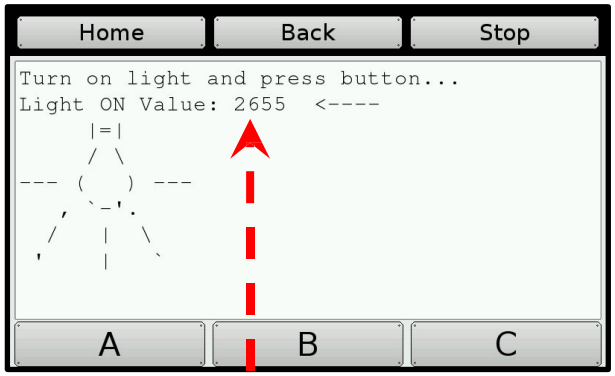
1. Wait for light.
2. Drive forward.
3. Wait for 3 seconds.
4. Stop motors.
5. End the program.

Comments

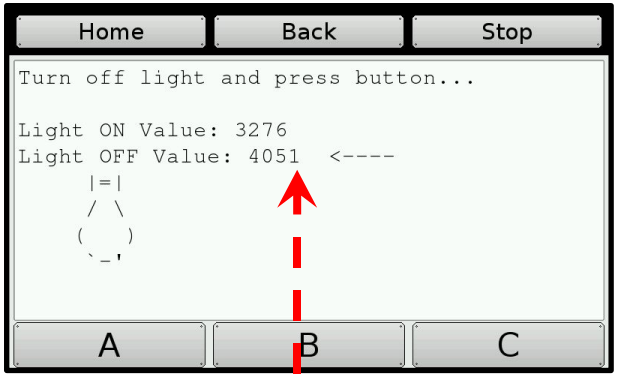
```
// 1. Wait for light.
// 2. Drive forward.
// 3. Wait for 3 seconds.
// 4. Stop motors.
// 5. End the program.
```

wait_for_light Calibration Routine

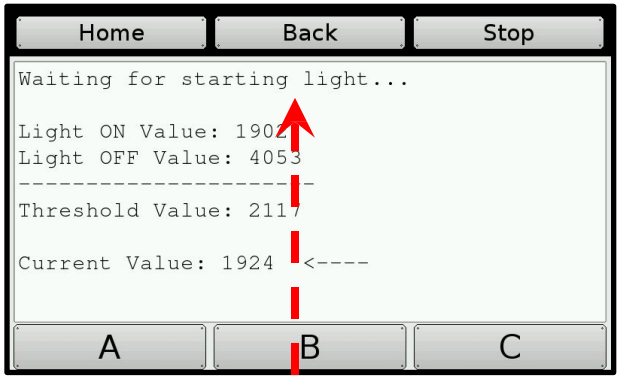
When you use the **wait_for_light ( )** function in your program, the following calibration routine will run automatically.



When the light is *on* (low value), press the "push" button.



When the light is *off* (high value), press the "push" button.



You will get a "Waiting for starting light" when done *correctly*. You will get a "BAD CALIBRATION" message when not done correctly, and you will need to push the "push" button to run through the routine again.

"push" button

Note: For Botball, **wait_for_light()** should be one of the first functions called in your program.

Starting with a Light

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Solution:

Pseudocode

1. Wait for light.
2. Drive forward.
3. Wait for 3 seconds.
4. Stop motors.
5. End the program.

Source Code

```
1  #include <kipr/wombat.h>
2
3  int main()
4  {
5      wait_for_light(0);
6
7      motor(0, 100); //forward
8      motor(3, 100);
9      msleep(3000);
10     ao();
11
12     return 0;
13 }
14
```

Execution: Compile and run your program (test it with a light sensor).

Starting with a light

KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Solution: Use a function!

Pseudocode

1. Wait for light.
2. Drive forward.
3. Wait for 3 seconds.
4. Stop motors.
5. End the program.

Execution: Compile and run your program.

Source Code

```
1  #include <kipr/wombat.h>
2
3  int main()
4  {
5      wait_for_light(0);
6
7      drive_forward();
8      msleep(3000);
9      ao();
10
11     return 0;
12 }
13 void drive_forward()
14 {
15     motor(0, 100);
16     motor(3, 100);
17 }
```

Starting Programs with a Light

The light sensor is a cool way to automatically start your robot and critical for Botball robots at the beginning of the game.

The wait_for_light() function allows your program to run after your robot senses a light.

> Note: It has a built-in calibration routine that will come up on the screen (a step-by-step guide for this calibration routine is on a following slide).

The light sensor senses infrared light, so light must be emitted from an incandescent light, not an LED light.

For our activities, you can use a flashlight.

The more light (infrared) detected, the lower the reported value.

wait_for_light Function

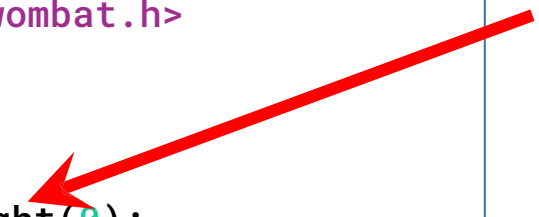
KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

```
wait_for_light(0);
// Waits for the light on port #0 before going to the next line.
```

Source Code

```
1  #include <kipr/wombat.h>
2
3  int main()
4  {
5      wait_for_light(0);
6      printf("I see the light!\n");
7      return 0;
8  }
9
10
```



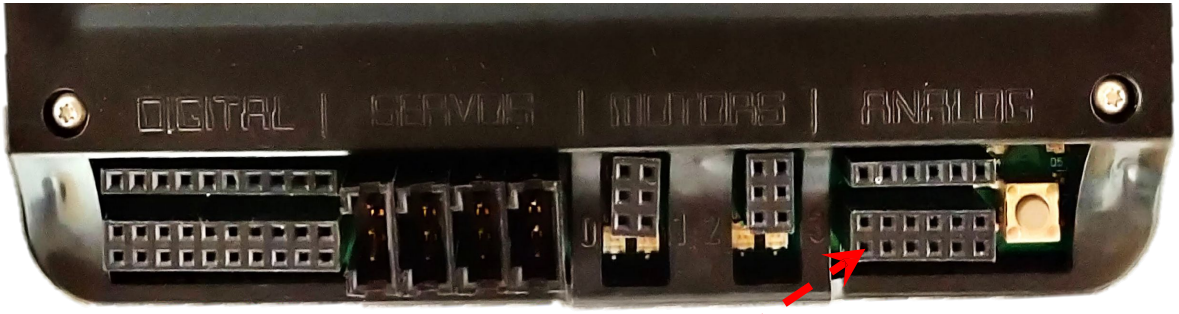
What is this?

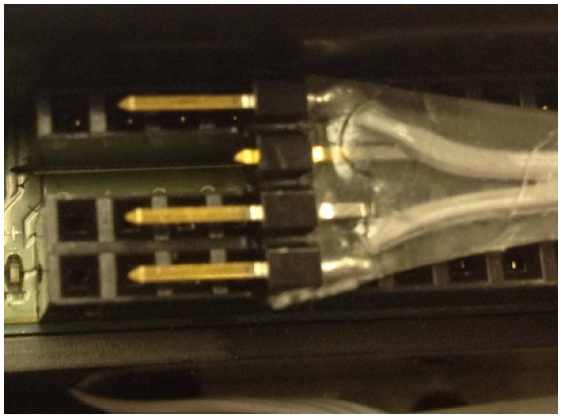
Plug in the Light Sensor

KISS INSTITUTE FOR PRACTICAL ROBOTICS

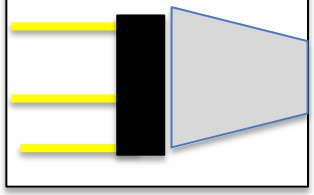
Botball®

(Light source needed, cell phone works)





Sensor plug orientation



Plug your light sensor into analog port 0

Starting with a Light

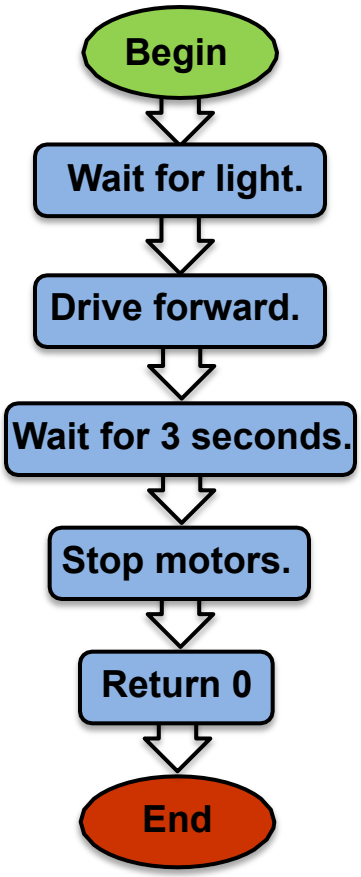
KISS INSTITUTE FOR PRACTICAL ROBOTICS

Botball®

Description: Write a program that waits for a light to come on, drives the DemoBot forward for 3 seconds, and then stops.

Analysis: What is the program supposed to do?

Flowchart



Pseudocode

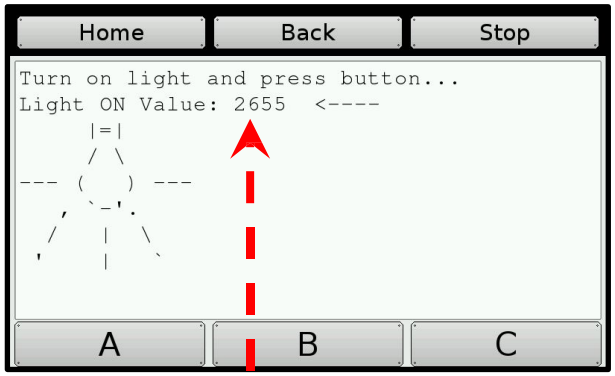
1. Wait for light.
2. Drive forward.
3. Wait for 3 seconds.
4. Stop motors.
5. End the program.

Comments

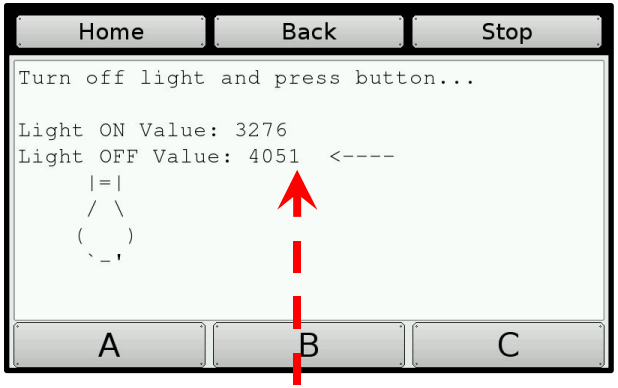
```
// 1. Wait for light.
// 2. Drive forward.
// 3. Wait for 3 seconds.
// 4. Stop motors.
// 5. End the program.
```

wait_for_light Calibration Routine

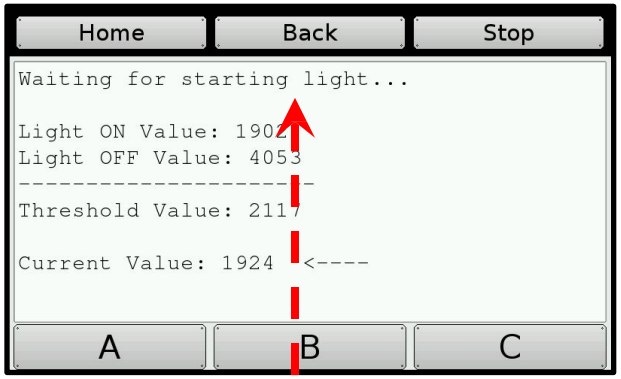
When you use the **wait_for_light( )**function in your program, the following calibration routine will run automatically.



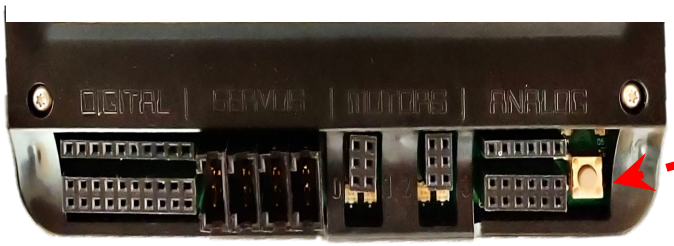
When the light is *on* (low value), press the "push" button.



When the light is *off* (high value), press the "push" button.



You will get a "Waiting for starting light" when done *correctly*. You will get a "BAD CALIBRATION" message when not done correctly, and you will need to push the "push" button to run through the routine again.



"push" button

Note: For Botball, **wait_for_light( )** should be one of the first functions called in your program.