# Light Sensor Library - Documentation

## Library Description:

The Light Sensor library uses a light sensor that changes its resistance based on how much light it detects. This resistance will be in the range of 3 k to 11 k ohms. It is used in a voltage divider with a 4.7 k ohm resistor and the voltage is read using a peripheral pin on the microcontroller. This value is analog so an analog to digital converter converts it into a voltage. This device is compatible with 3.0 V and 3.3 V sources. Call the function lightDetected() to check for light value.

## Hardware Description:

Microcontroller: PIC24FJ64GA002 | Microchip Technology
Photocell light sensor with 3k-11k ohm range. The one found under the link is 1k-10k but would work. The one used in this device was found in the ECE depot. Photocell light sensor

## Full Documentation:

### void initLightSensor()

- No arguments, no output
- Description: The function will initialize the analog to digital conversion and assign it to timer 3. It will also assign pin RA0 (pin 2) to input.

### void initBuffer()

- No arguments, no output
- Description: Initializes buffer to the ideal size, 10 is default.

### void putVal(int ADCvalue)

- Argument is the ADCvalue from the analog to digital buffer, no output
- Description: Fills created buffer with ADC values

### int getAvg()

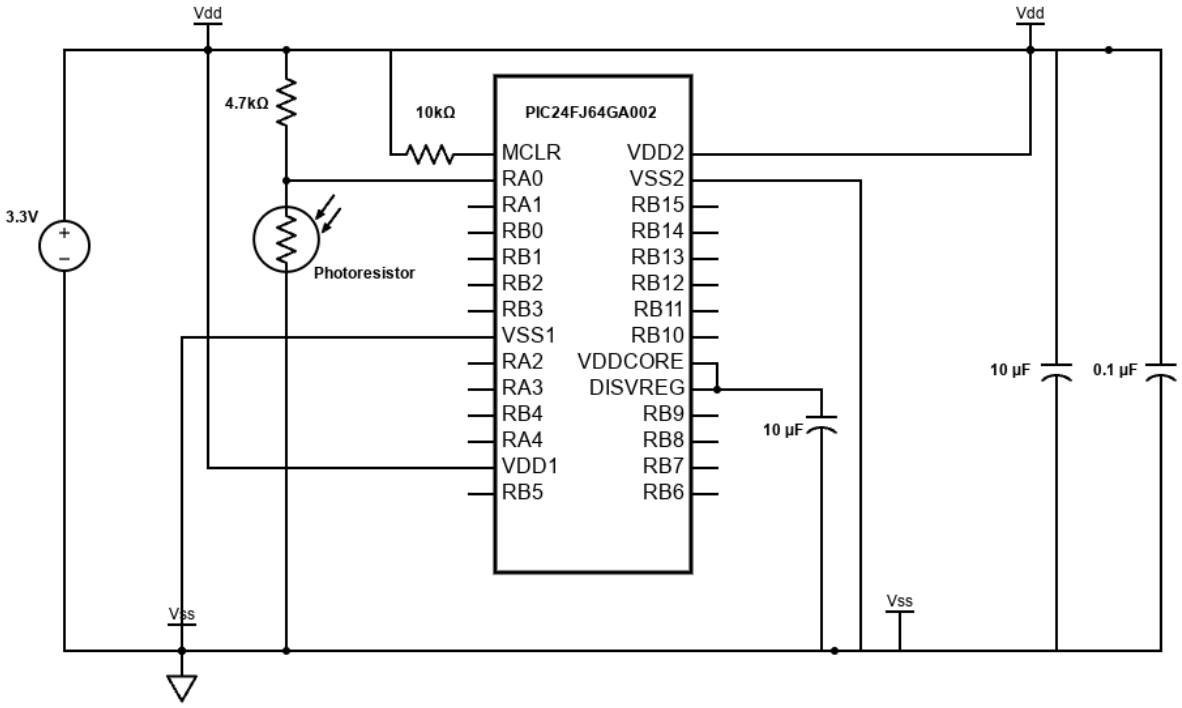
- No arguments, outputs the average as an integer
- Description: The function will find the average of all the ADC values in the buffer over the sampling period.

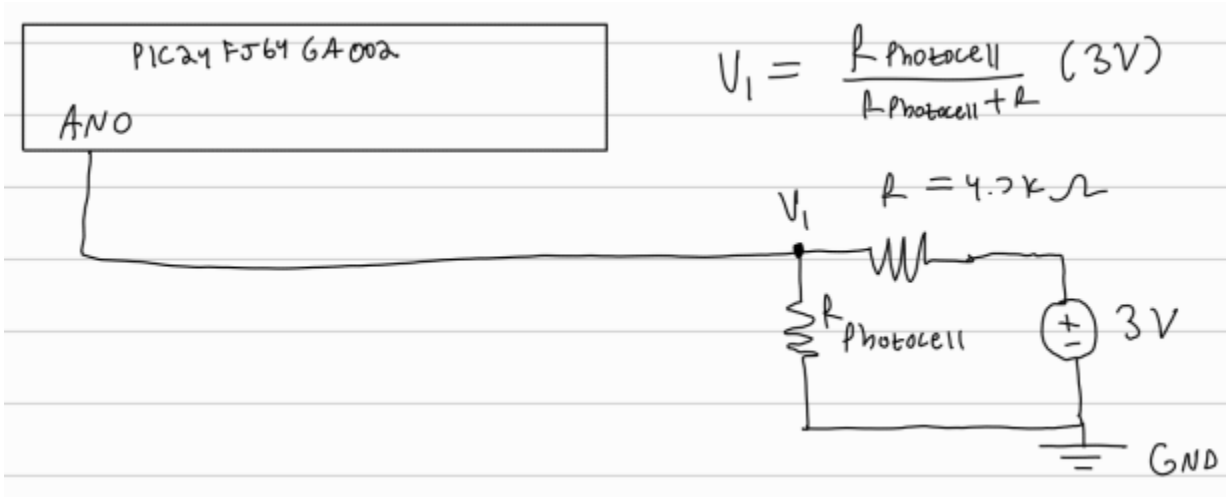
## int lightDetected()

- No arguments, outputs integer to state if it is light or dark.
- Description: The function will check the value of the getAvg function and determine if the light detected is above the threshold. Will return -1 if the buffer is not yet full, 0 if it is dark, and 1 if light is detected.

## Basic Usage Example:

### PIC24FJ64GA002 Circuit Schematic:



In the main code, include the “xc.h” library. Additionally, ensure to include the necessary flash configuration words into the main file (see Section 24.1 of the PIC24 Family Reference Manual for details).

**Main Code Example:**

```
// CW1: FLASH CONFIGURATION WORD 1 (see PIC24 Family Reference Manual 24.1)
#pragma config ICS = PGx1          // Comm Channel Select (Emulator EMUC1/EMUD1 pins are shared with PGC1/PGD1)
#pragma config FWDTEN = OFF        // Watchdog Timer Enable (Watchdog Timer is disabled)
#pragma config GWRP = OFF          // General Code Segment Write Protect (Writes to program memory are allowed)
#pragma config GCP = OFF           // General Code Segment Code Protect (Code protection is disabled)
#pragma config JTAGEN = OFF        // JTAG Port Enable (JTAG port is disabled)

// CW2: FLASH CONFIGURATION WORD 2 (see PIC24 Family Reference Manual 24.1)
#pragma config I2C1SEL = PRI       // I2C1 Pin Location Select (Use default SCL1/SDA1 pins)
#pragma config IOL1WAY = OFF       // IOLOCK Protection (IOLOCK may be changed via unlocking seq)
#pragma config OSCIOFNC = ON       // Primary Oscillator I/O Function (CLKO/RC15 functions as I/O pin)
#pragma config FCKSM = CSECME      // Clock Switching and Monitor (Clock switching is enabled,
                                   // Fail-Safe Clock Monitor is enabled)
#pragma config FNOSC = FRCPLL      // Oscillator Select (Fast RC Oscillator with PLL module (FRCPLL))
```

```
#include "xc.h"
#include "LightSensor.h"

int main(void) {
        initLightSensor();
        Int lightSensed = lightDetected(); // Determine if light detected is above set
// threshold. Will return 1 if it is and 0 if it is darker.
        return 0;
}
```

## Advanced Usage Example:

The Light Sensor library is useful for detecting light and enabling another device based on the light exposure. The below example shows how to check if light is detected or not and then set off an alarm if the light is above the threshold. The alarm will then be set off if the light falls below the threshold.

```
// CW1: FLASH CONFIGURATION WORD 1 (see PIC24 Family Reference Manual 24.1)
#pragma config ICS = PGx1          // Comm Channel Select (Emulator EMUC1/EMUD1 pins are shared with PGC1/PGD1)
#pragma config FWDTEN = OFF        // Watchdog Timer Enable (Watchdog Timer is disabled)
#pragma config GWRP = OFF          // General Code Segment Write Protect (Writes to program memory are allowed)
#pragma config GCP = OFF           // General Code Segment Code Protect (Code protection is disabled)
#pragma config JTAGEN = OFF        // JTAG Port Enable (JTAG port is disabled)

// CW2: FLASH CONFIGURATION WORD 2 (see PIC24 Family Reference Manual 24.1)
#pragma config I2C1SEL = PRI       // I2C1 Pin Location Select (Use default SCL1/SDA1 pins)
#pragma config IOL1WAY = OFF       // IOLOCK Protection (IOLOCK may be changed via unlocking seq)
#pragma config OSCIOFNC = ON       // Primary Oscillator I/O Function (CLKO/RC15 functions as I/O pin)
```

```
#pragma config FCKSM = CSECME      // Clock Switching and Monitor (Clock switching is enabled,
                                   // Fail-Safe Clock Monitor is enabled)
#pragma config FNOSC = FRCPLL      // Oscillator Select (Fast RC Oscillator with PLL module (FRCPLL))

#include "xc.h"
#include "LightSensor.h"
#include "Alarm.h"

void setup();
void loop();

int main(void) {
    setup();
    loop();
}

void setup() {
    initLightSensor();
    initAlarm();
}

void loop() {
    while(1) {
        if(lightDetected()) {
            turnOnAlarm; // light detected, set off alarm
}
            while(lightDetected()); //alarm stays on
            turnOffAlarm; // light is gone, turn off alarm
}
```