

list5.txt

入力:VL53L0X.hが見つかりません。かわりに、Adafruit\_VL53L0X.hは使えませんか？

/\*!

\* @file Adafruit\_VL53L0X.h

This is a library for the Adafruit VL53L0X Sensor Breakout

Designed specifically to work with the VL53L0X sensor from Adafruit

----> <https://www.adafruit.com/products/3317>

These sensors use I2C to communicate, 2 pins are required to interface

Adafruit invests time and resources providing this open source code, please support Adafruit and open-source hardware by purchasing products from Adafruit!

Written by Limor Fried/Ladyada for Adafruit Industries.

Updated by Andrew DeVries for Digital Example to include methods needed for Interrupt triggering.

BSD license, all text above must be included in any redistribution

\*\*\*\*\*/

```
#ifndef ADAFRUIT_VL53L0X_H
#define ADAFRUIT_VL53L0X_H
```

```
#if (ARDUINO >= 100)
#include "Arduino.h"
#else
#include "WProgram.h"
#endif
```

```
#include "Wire.h"
#include "vl53l0x_api.h"
```

```
#define VL53L0X_I2C_ADDR 0x29 ///< Default sensor I2C address
```

```
*****/
```

```
/*!
```

```
  @brief Class that stores state and functions for interacting with VL53L0X time-of-flight sensor chips
```

```
*/
```

```
*****/
```

```
class Adafruit_VL53L0X {
public:
  /** Sensor configurations */
  typedef enum {
    VL53L0X_SENSE_DEFAULT = 0,
    VL53L0X_SENSE_LONG_RANGE,
    VL53L0X_SENSE_HIGH_SPEED,
    VL53L0X_SENSE_HIGH_ACCURACY
  } VL53L0X_Sense_config_t;
```

```

list5.txt
boolean begin(uint8_t i2c_addr = VL53L0X_I2C_ADDR, boolean debug = false,
              TwoWire *i2c = &Wire,
              VL53L0X_Sense_config_t vl_config = VL53L0X_SENSE_DEFAULT);
boolean setAddress(uint8_t newAddr);

// uint8_t getAddress(void); // not currently implemented

/*****
/*!
  @brief get a ranging measurement from the device
  @param pRangingMeasurementData the pointer to the struct the data will be
  stored in
  @param debug Optional debug flag. If true debug information will print via
  Serial.print during execution. Defaults to false.
  @returns True if address was set successfully, False otherwise
*/
*****/
VL53L0X_Error
rangingTest(VL53L0X_RangingMeasurementData_t *pRangingMeasurementData,
            boolean debug = false) {
    return getSingleRangingMeasurement(pRangingMeasurementData, debug);
};

VL53L0X_Error getSingleRangingMeasurement(
    VL53L0X_RangingMeasurementData_t *pRangingMeasurementData,
    boolean debug = false);
void
printRangeStatus(VL53L0X_RangingMeasurementData_t *pRangingMeasurementData);

VL53L0X_Error getRangingMeasurement(
    VL53L0X_RangingMeasurementData_t *pRangingMeasurementData,
    boolean debug = false);
VL53L0X_Error startMeasurement(boolean debug = false);
VL53L0X_Error stopMeasurement(boolean debug = false);
VL53L0X_Error getLimitCheckCurrent(uint8_t LimitCheckId,
                                   FixPoint1616_t *pLimitCheckCurrent,
                                   boolean debug = false);
VL53L0X_Error getDeviceMode(VL53L0X_DeviceModes *pDeviceMode,
                             boolean debug = false);
VL53L0X_Error setDeviceMode(VL53L0X_DeviceModes DeviceMode,
                             boolean debug = false);

VL53L0X_Error setInterruptThresholds(FixPoint1616_t ThresholdLow,
                                     FixPoint1616_t ThresholdHigh,
                                     boolean debug = false);
VL53L0X_Error getInterruptThresholds(FixPoint1616_t *pThresholdLow,
                                     FixPoint1616_t *pThresholdHigh,
                                     boolean debug = false);
VL53L0X_Error clearInterruptMask(boolean debug = false);

VL53L0X_Error getGpioConfig(VL53L0X_DeviceModes *pDeviceMode,
                            VL53L0X_GpioFunctionality *pFunctionality,

```

```

list5.txt
VL53L0X_InterruptPolarity *pPolarity,
boolean debug = false);
VL53L0X_Error setGpioConfig(VL53L0X_DeviceModes DeviceMode,
VL53L0X_GpioFunctionality Functionality,
VL53L0X_InterruptPolarity Polarity,
boolean debug = false);

VL53L0X_Error Status =
    VL53L0X_ERROR_NONE; ///< indicates whether or not the sensor has
                        ///< encountered an error
// Add similar methods as Adafruit_VL6180X class adapted to range of device
uint16_t readRange(void);
// float readLux(uint8_t gain);
uint8_t readRangeStatus(void);

boolean startRange(void);
boolean isRangeComplete(void);
boolean waitRangeComplete(void);
uint16_t readRangeResult(void);

boolean startRangeContinuous(uint16_t period_ms = 50);
void stopRangeContinuous(void);

// void setTimeout(uint16_t timeout) { io_timeout = timeout; }
// uint16_t getTimeout(void) { return io_timeout; }
/*****/
/*!
    @brief timeout status
    @returns True if timeout has occurred, False otherwise
*/
/*****/
boolean timeoutOccurred(void) { return false; }

boolean configSensor(VL53L0X_Sense_config_t vl_config);

// Export some wrappers to internal setting functions
// that are used by the above helper function to allow
// more complete control.
boolean setMeasurementTimingBudgetMicroSeconds(uint32_t budget_us);
uint32_t getMeasurementTimingBudgetMicroSeconds(void);

boolean setVcselPulsePeriod(VL53L0X_VcselPeriod VcselPeriodType,
uint8_t VCSELPulsePeriod);

uint8_t getVcselPulsePeriod(VL53L0X_VcselPeriod VcselPeriodType);

boolean setLimitCheckEnable(uint16_t LimitCheckId, uint8_t LimitCheckEnable);
uint8_t getLimitCheckEnable(uint16_t LimitCheckId);
boolean setLimitCheckValue(uint16_t LimitCheckId,
FixPoint1616_t LimitCheckValue);
FixPoint1616_t getLimitCheckValue(uint16_t LimitCheckId);

```

list5.txt

```
private:
  VL53L0X_Dev_t MyDevice;
  VL53L0X_Dev_t *pMyDevice = &MyDevice;
  VL53L0X_DeviceInfo_t DeviceInfo;

  uint8_t _rangeStatus;
};

#endif
(コピペ部分おわり)
```

GPT:はい、'Adafruit\_VL53L0X.h' ライブラリを使用しても問題ありません。このライブラリを使用すると、VL53L0Xセンサから距離データを簡単に読み取ることができます。

以下は、Raspberry Pi PicoでArduino IDEを使用し、'Adafruit\_VL53L0X' ライブラリを利用して距離データをシリアルで出力するための基本的なコード例です。

```
#include <Wire.h>
#include <Adafruit_VL53L0X.h>

Adafruit_VL53L0X lox = Adafruit_VL53L0X();

void setup() {
  Serial.begin(115200);
  Wire.begin();

  if (!lox.begin()) {
    Serial.println(F("Failed to boot VL53L0X"));
    while(1);
  }
}

void loop() {
  VL53L0X_RangingMeasurementData_t measure;

  lox.rangingTest(&measure, false); // pass in 'true' to get debug data
  printout!

  if (measure.RangeStatus != 4) { // phase failures have incorrect data
    Serial.print("Distance (mm): "); Serial.println(measure.RangeMilliMeter);
  } else {
    Serial.println("Out of range");
  }

  delay(100);
}
```

このコードは、センサから距離を読み取り、シリアルモニタに出力します。ただし、'Adafruit\_VL53L0X'ライブラリをArduino IDEにインストールする必要があります。また、シリアルモニタのボーレートを115200に設定してください。

ライブラリのインストールは、Arduino IDEの「スケッチ」メニューの「ライブラリをインクルード」→「ライブラリを管理...」から検索してインストールできます。