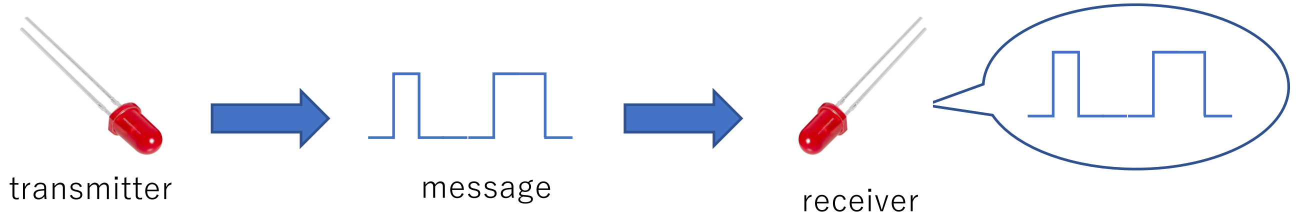
# Weak Signal Detection in Visible Light Communication

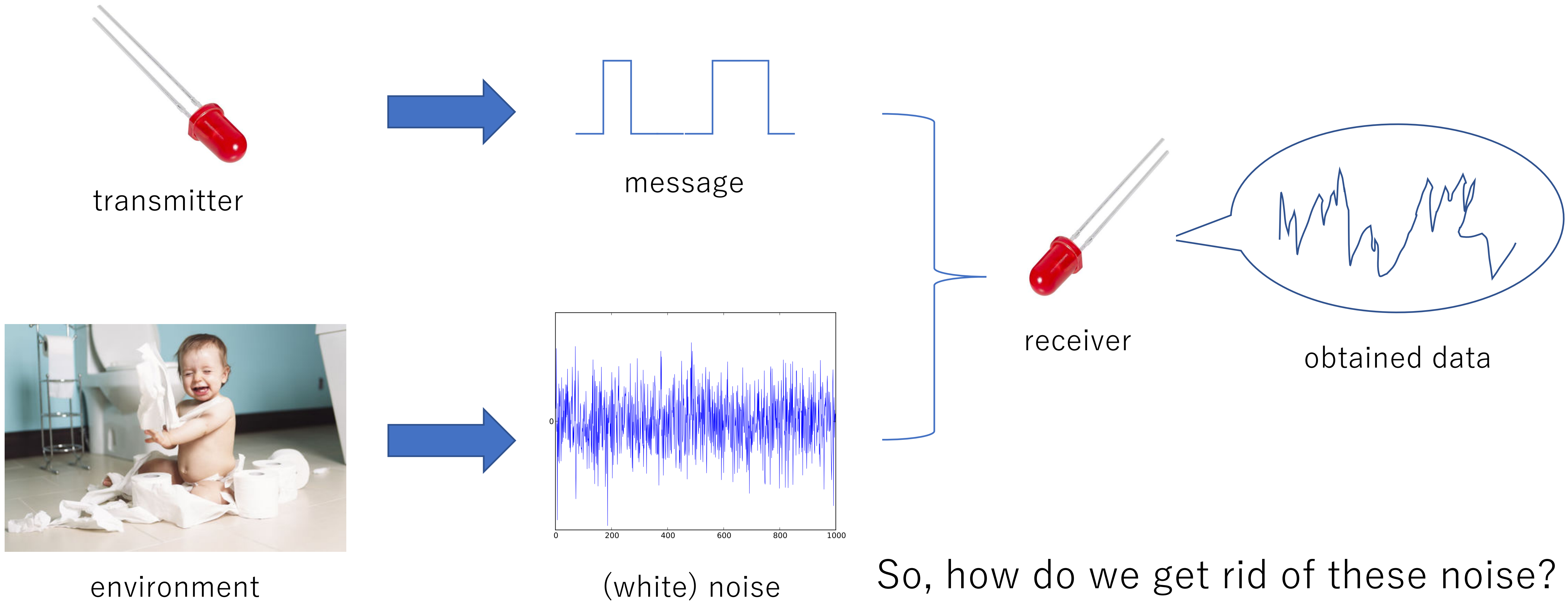
Go Nishimura

## VLC and the noise

Ideal visual light communication(VLC)



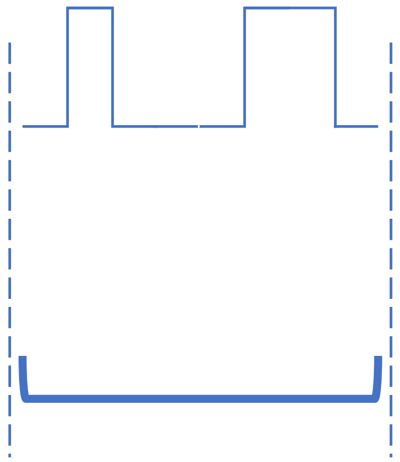
But in real...



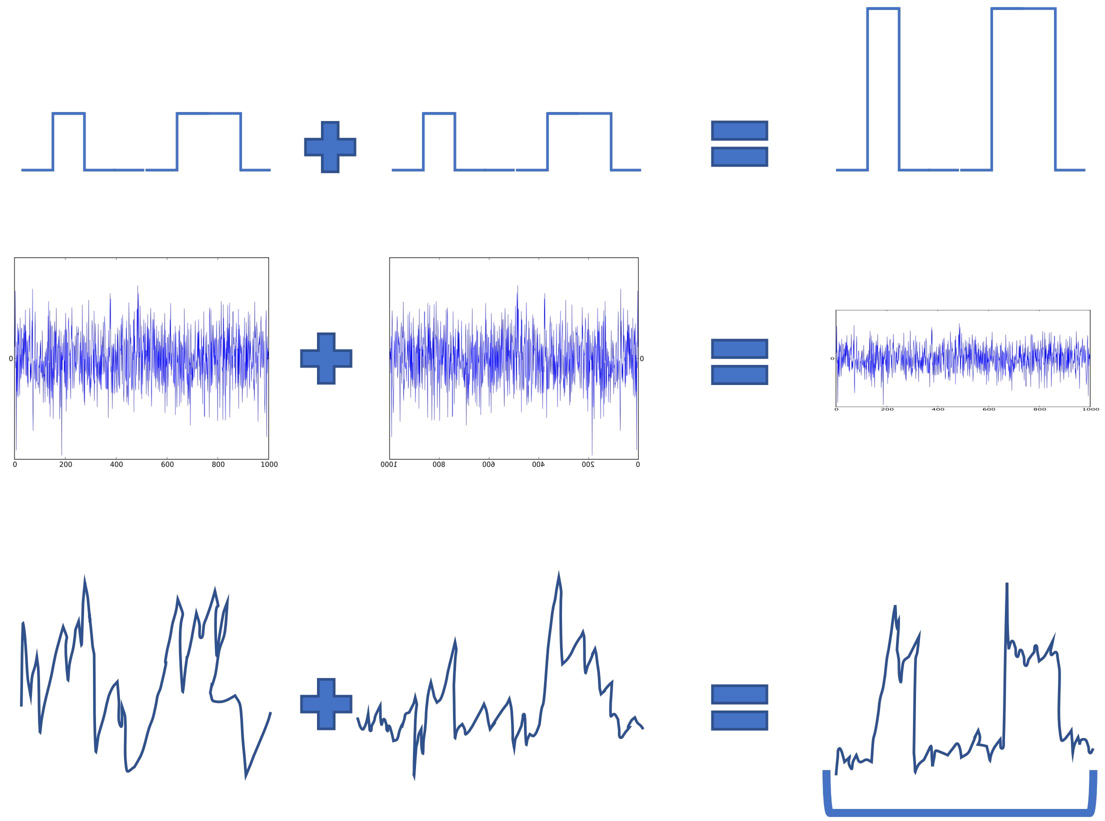
So, how do we get rid of these noise?

# Basic concept: get signals again and again!

1: Make a data frame whose size is same as the message



2: Send and receive data again and again into data frame until the message become significant



Since the same message is repeated, the difference of 0 and 1 get more big.

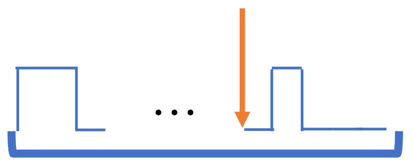
If you add up white noise, each different patterns erase each other and end up in small difference of 0 and 1.

Same data frame add up data ending up in more significant message and small noise.

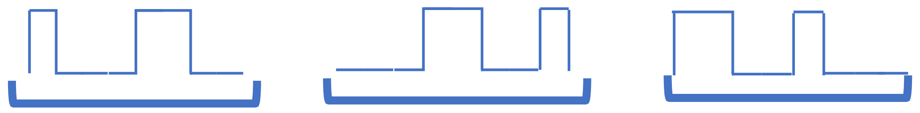
# Goal: Find already-taught sign from the data frame

- Since I couldn’t buy RTC, my mission was to read key message from the data frame.
- That’s because the transmitter and the receiver can’t synchronize the time of the start of the message.
- In serial communication, a message have it’s sign of start(STX) and end(ETX) at the top/bottom of the message.
- So, if we can identify an already-taught sign, we can read the message correctly

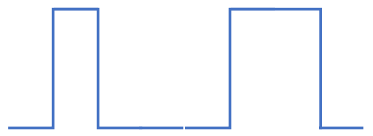
In future…



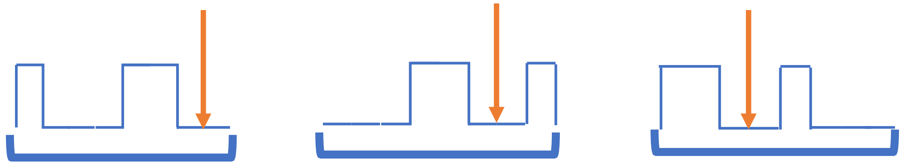
Since no RTC, data frame may look like this:



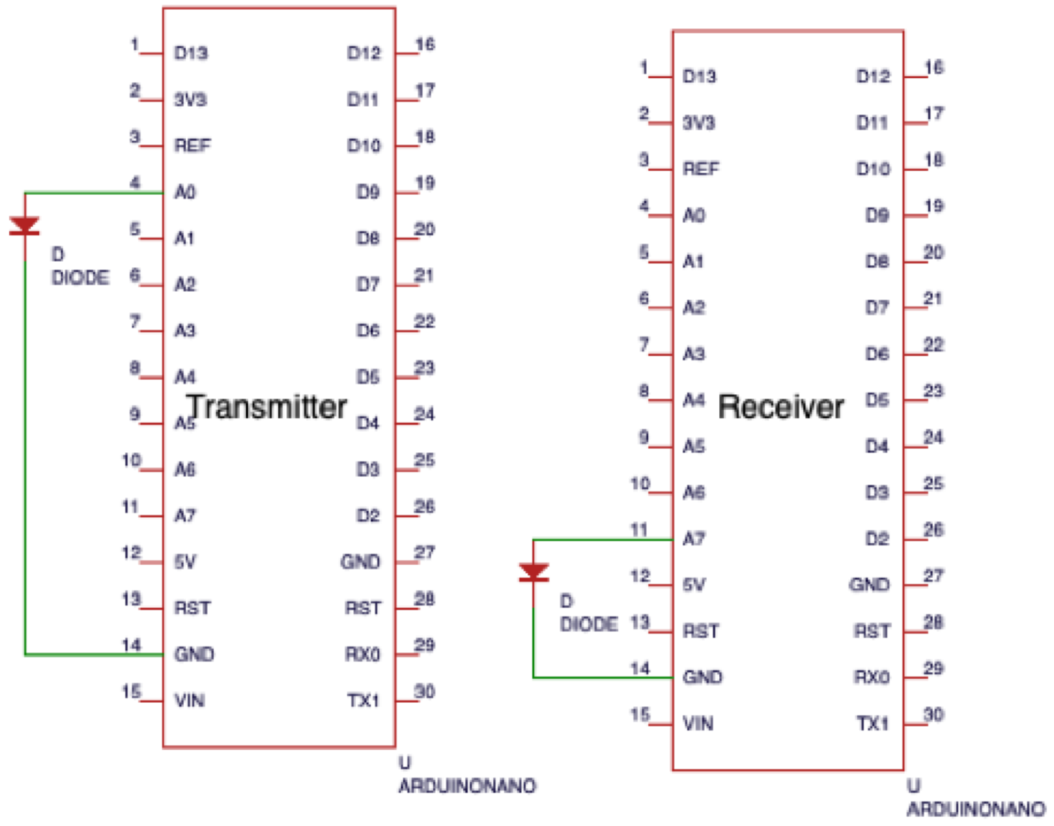
But if we knew the start sign,

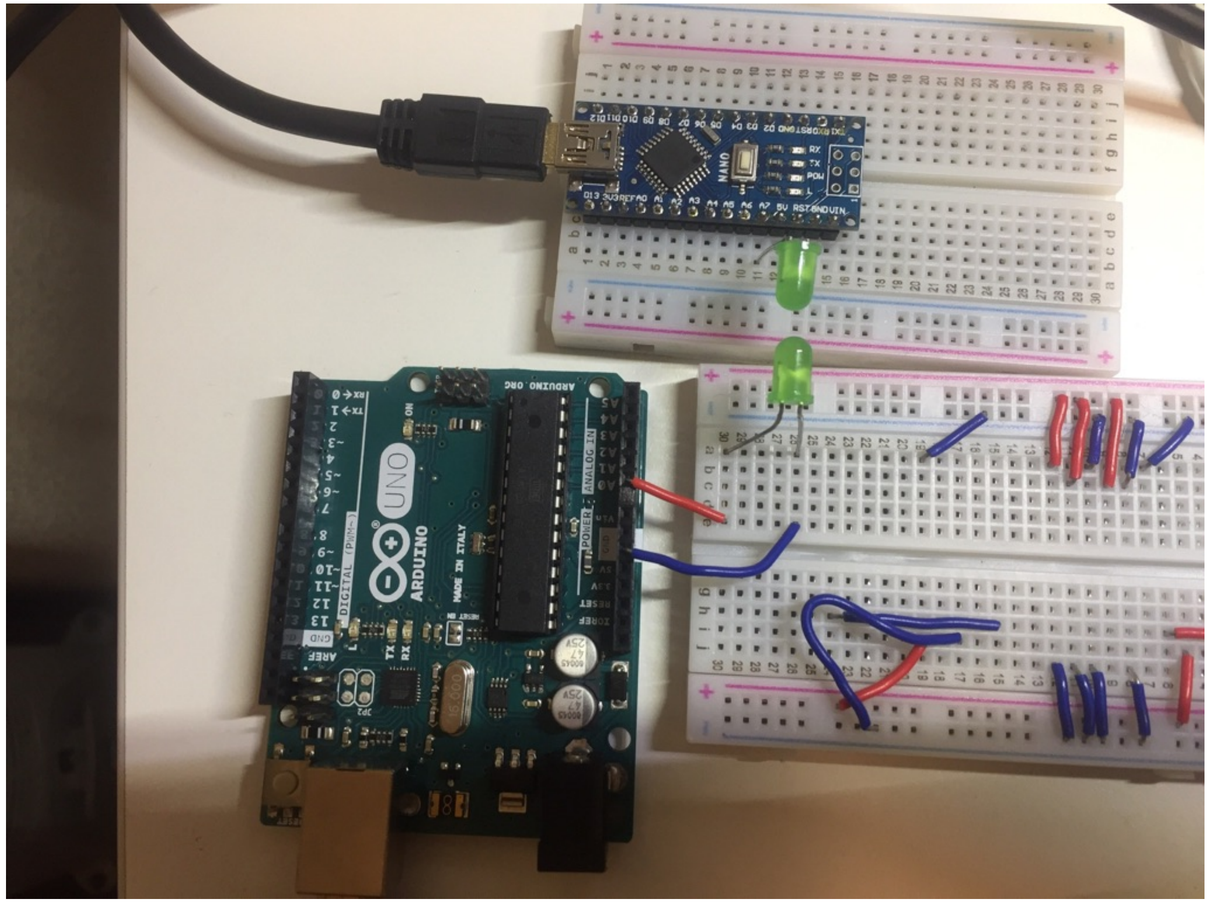


we know from where to read message



# Circuit diagram

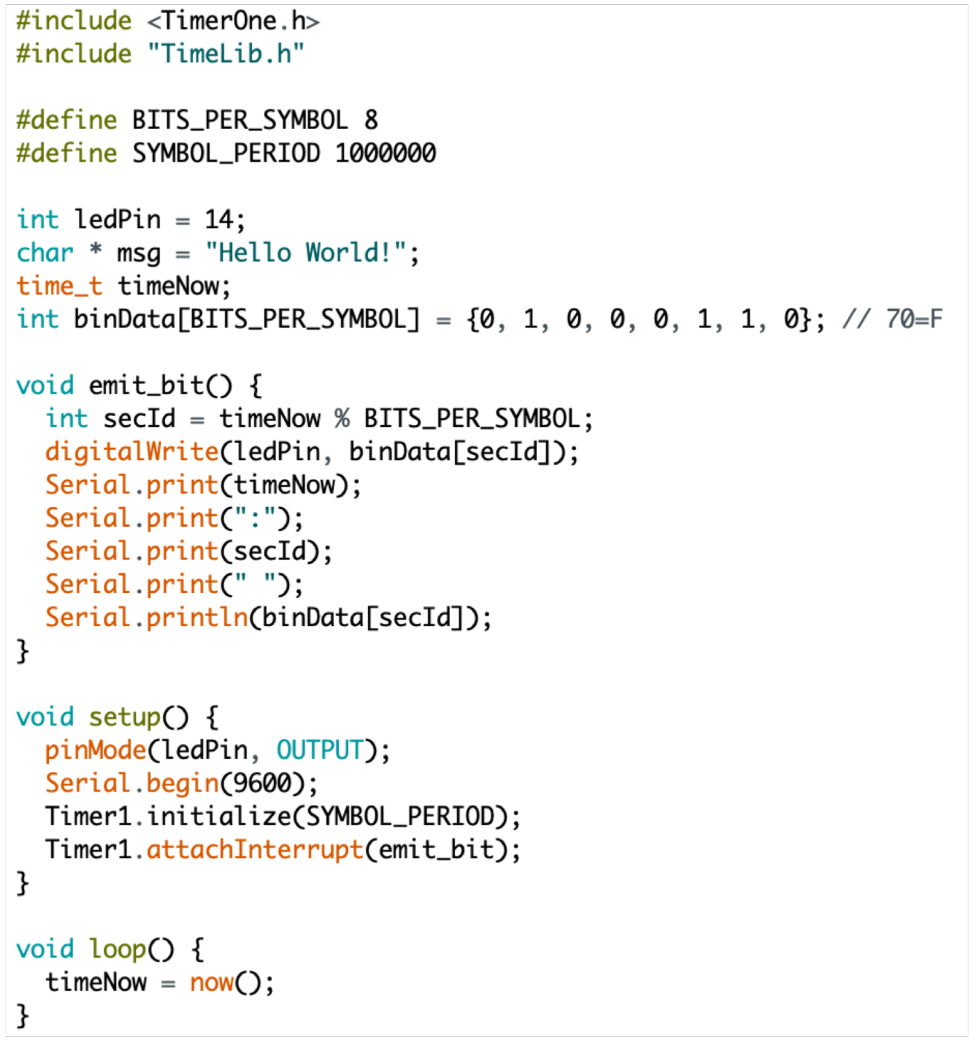




# Program (transmitter)

weakSigTx.ino

turn on or off the LED every 1 sec

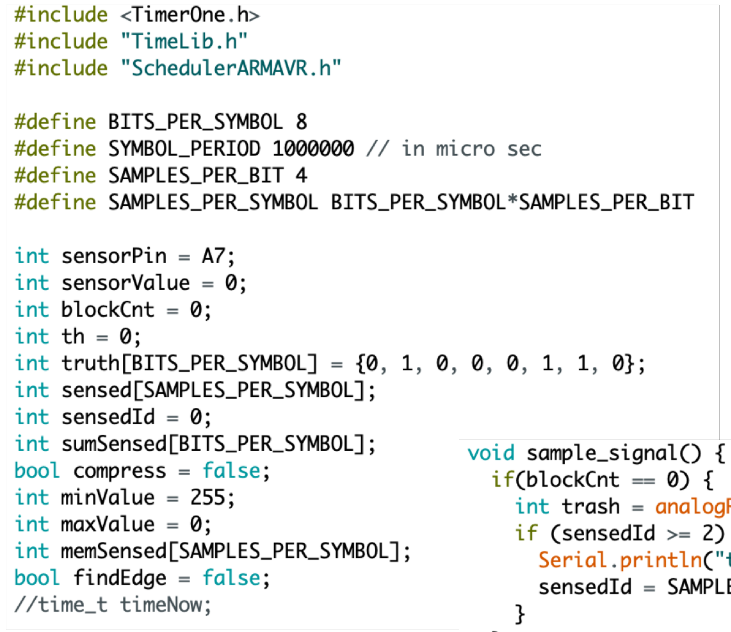


```
#include <TimerOne.h>
#include "TimeLib.h"

#define BITS_PER_SYMBOL 8
#define SYMBOL_PERIOD 1000000

int ledPin = 14;
char * msg = "Hello World!";
time_t timeNow;
int binData[BITS_PER_SYMBOL] = {0, 1, 0, 0, 0, 1, 1, 0}; // 70=F

void emit_bit() {
  int secId = timeNow % BITS_PER_SYMBOL;
  digitalWrite(ledPin, binData[secId]);
  Serial.print(timeNow);
  Serial.print(":");
  Serial.print(secId);
  Serial.print(" ");
  Serial.println(binData[secId]);
}

void setup() {
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
  Timer1.initialize(SYMBOL_PERIOD);
  Timer1.attachInterrupt(emit_bit);
}

void loop() {
  timeNow = now();
}
```

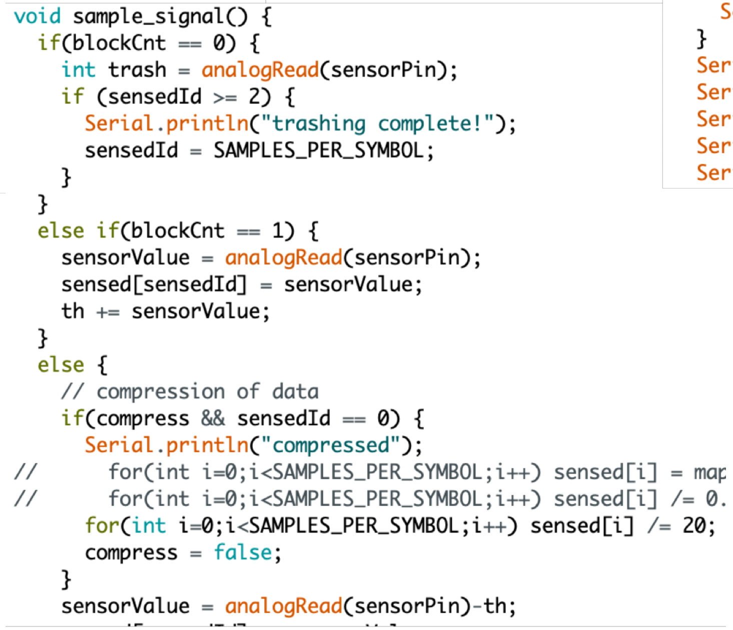
# Program (Receiver)

weakSigRx.ino

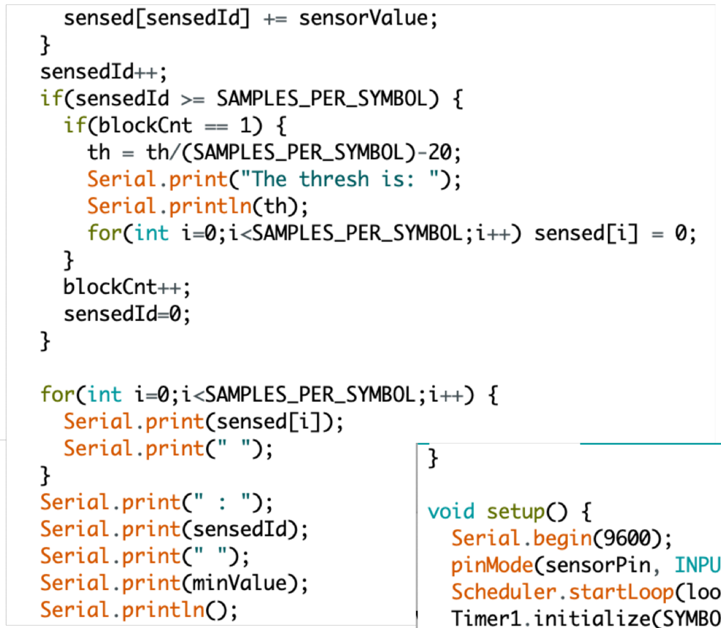
… too long!



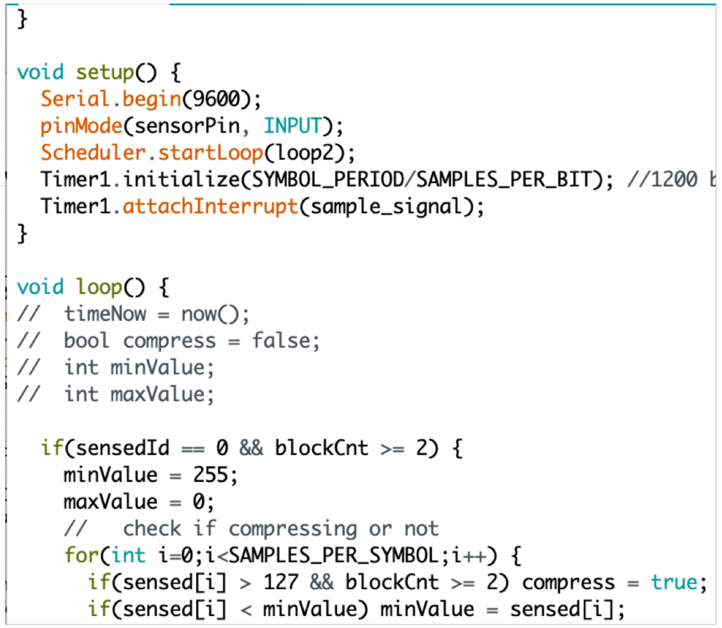
```
#include <TimerOne.h>
#include "TimeLib.h"
#include "SchedulerARMAVR.h"

#define BITS_PER_SYMBOL 8
#define SYMBOL_PERIOD 1000000 // in micro sec
#define SAMPLES_PER_BIT 4
#define SAMPLES_PER_SYMBOL BITS_PER_SYMBOL*SAMPLES_PER_BIT

int sensorPin = A7;
int sensorValue = 0;
int blockCnt = 0;
int th = 0;
int truth[BITS_PER_SYMBOL] = {0, 1, 0, 0, 0, 1, 1, 0};
int sensed[SAMPLES_PER_SYMBOL];
int sensedId = 0;
int sumSensed[BITS_PER_SYMBOL];
bool compress = false;
int minValue = 255;
int maxValue = 0;
int memSensed[SAMPLES_PER_SYMBOL];
bool findEdge = false;
//time_t timeNow;
```



```
void sample_signal() {
  if(blockCnt == 0) {
    int trash = analogRead(sensorPin);
    if (sensedId >= 2) {
      Serial.println("trashing complete!");
      sensedId = SAMPLES_PER_SYMBOL;
    }
  }
  else if(blockCnt == 1) {
    sensorValue = analogRead(sensorPin);
    sensed[sensedId] = sensorValue;
    th += sensorValue;
  }
  else {
    // compression of data
    if(compress && sensedId == 0) {
      Serial.println("compressed");
//      for(int i=0;i<SAMPLES_PER_SYMBOL;i++) sensed[i] = map
//      for(int i=0;i<SAMPLES_PER_SYMBOL;i++) sensed[i] /= 0.
      for(int i=0;i<SAMPLES_PER_SYMBOL;i++) sensed[i] /= 20;
      compress = false;
    }
    sensorValue = analogRead(sensorPin)-th;
```



```
    sensed[sensedId] += sensorValue;
  }
  sensedId++;
  if(sensedId >= SAMPLES_PER_SYMBOL) {
    if(blockCnt == 1) {
      th = th/(SAMPLES_PER_SYMBOL)-20;
      Serial.print("The thresh is: ");
      Serial.println(th);
      for(int i=0;i<SAMPLES_PER_SYMBOL;i++) sensed[i] = 0;
    }
    blockCnt++;
    sensedId=0;
  }

  for(int i=0;i<SAMPLES_PER_SYMBOL;i++) {
    Serial.print(sensed[i]);
    Serial.print(" ");
  }
  Serial.print(" : ");
  Serial.print(sensedId);
  Serial.print(" ");
  Serial.print(minValue);
  Serial.println();
```



```
}

void setup() {
  Serial.begin(9600);
  pinMode(sensorPin, INPUT);
  Scheduler.startLoop(loop2);
  Timer1.initialize(SYMBOL_PERIOD/SAMPLES_PER_BIT); //1200
  Timer1.attachInterrupt(sample_signal);
}

void loop() {
//  timeNow = now();
//  bool compress = false;
//  int minValue;
//  int maxValue;

  if(sensedId == 0 && blockCnt >= 2) {
    minValue = 255;
    maxValue = 0;
    //   check if compressing or not
    for(int i=0;i<SAMPLES_PER_SYMBOL;i++) {
      if(sensed[i] > 127 && blockCnt >= 2) compress = true;
      if(sensed[i] < minValue) minValue = sensed[i];
```

and more…

Let’s try!