Electronic Supplementary Information (ESI)

Telechemistry: monitoring chemical reactions *via* the cloud using the Particle Photon Wi-Fi module

Gurpur Rakesh D. Prabhu[a,b], Henryk A. Witek[a,c]* and Pawel L. Urban[b,d]*

[a] *Department of Applied Chemistry, National Chiao Tung University*

1001 University Rd., Hsinchu, 30010, Taiwan.

[b] *Department of Chemistry, National Tsing Hua University*

101, Section 2, Kuang-Fu Rd., Hsinchu, 30013, Taiwan.

[c] *Center for Emergent Functional Matter Science, National Chiao Tung University*

1001 University Rd., Hsinchu 30010, Taiwan.

[d] *Frontier Research Center on Fundamental and Applied Sciences of Matters,*

National Tsing Hua University

101, Section 2, Kuang-Fu Rd., Hsinchu, 30013, Taiwan.

* Corresponding authors:

H.A. Witek, e-mail: hwitek@mail.nctu.edu.tw

P.L. Urban, e-mail: urban@mx.nthu.edu.tw, web: http://www.UrbanLab.tw

PARTICLE PHOTON CODE

```
// Library for DHT11 sensor
#include <Adafruit_DHT.h>

// Define pins
#define DHTPIN 2     // DHT11 data pin connected to
#define DHTTYPE DHT11              // DHT sensor type
DHT dht(DHTPIN, DHTTYPE);
int button = D0;//Button to indicate the start of the experiment
int pHelectrode = A5;//analog pin for pH sensor
int spillsensor = A1;//analog pin for spillage sensor
int pHelectrodePower = A0;//power supply to pH sensor
int pst = D6;//digital pin for power switch
int val = 0;
int raw = 0;
int counter = 0;
int average = 0;
float tem[21];//array for 21 temperature values
float humis[21];//array for 21 humidity values
int numberOfReads = 21;//median filter data sample size
int pHelectrodeCheckTime = 1;//in milliseconds
//pHelectrodeCheckTime*counter*DHT11_Sampling interval*median filter size = data transmission frequency
int lastpHelectrodeReport;// this records the time of reporting the pH
char pHValue[4];//pH string
float pH;//define pH
char humi[4];//humidity string
char temp[4];//temperature string
char spill[4];//spillage string

//Tell Photon if the pins are inputs or outputs
void setup() {
  pinMode(pHelectrode, INPUT);
  pinMode(pHelectrodePower, OUTPUT);
  pinMode(button, INPUT);
  pinMode(pst, OUTPUT);
  digitalWrite(pst, HIGH);//power switch on
  digitalWrite(pHelectrodePower, HIGH);//pHelectrode on
  Particle.function("switch",switchToggle);//power witch on/off
  lastpHelectrodeReport = millis();// Initialize the timers using millis, the Photon's internal clock
  dht.begin();//DHT11 sensor on
  //publish upon powering the photon (e.g. pH 1; temperature 1; humidity 1; spill 1)
  if (pHelectrodePower = HIGH) {
      val = digitalRead(button);
      sprintf(pHValue,"%d", val);
      Particle.publish("pH", pHValue, 60, PRIVATE);
      sprintf(spill,"%d", val);
      Particle.publish("Spill", spill, 60, PRIVATE);
      sprintf(temp, "%d", val);
      Particle.publish("Temperature / °C", temp, 60, PRIVATE);
      sprintf(humi, "%d", val);
      Particle.publish("Humidity / %", humi, 60, PRIVATE);
  }
}
```

```
void loop() {
    //check if it is time to report the sensor values to the Particle web site
  if ((millis() - lastpHelectrodeReport) >= pHelectrodeCheckTime) {
      raw += analogRead(pHelectrode);//sum of pH values aquired
      counter++;
      lastpHelectrodeReport = millis();//update current time for the next time through the loop
      if (counter == 2000) {
        counter = 0;
        average = raw/2000;//average 2000 acqired pH values
        //pH = average;//activate the comment during pH calibration
        //obtain sensor values for pH calibration buffers (pH 7, pH 4 and pH 10)
        //determine pH calibration equation
        //uncomment next line during pH calibration and substitute new calibration equation
        pH = (average + 1497)/431.07;//pH calibration equation
        sprintf(pHValue,"%.2f", pH);//convert float to a string with 2 digitis after decimal point
        Particle.publish("pH", pHValue, 60, PRIVATE);// publish pH value
        raw = 0;
        average = 0;
        for (int i=0; i<numberOfReads; i++) {
            tem[i] = dht.getTempCelcius(); // fill the array with 21 readings, Read temperature as Celsius
            humis[i] = dht.getHumidity();// fill the array with 21 readings, Read humidity as %
            delay(3000);//sampling interval
        }
        qsort(tem, 21, sizeof(float), compare);//sort to put any outliers on either end of the array
        float t = (tem[6]+tem[7]+tem[8]+tem[9]+tem[10]+tem[11]+tem[12]+tem[13]+tem[14]+tem[15]+tem[16])/11;//average center values
        sprintf(temp, "%.1f", t);//convert float to a string with 2 digitis after decimal point
        if (t <= 40) Particle.publish("Temperature / °C", temp, 60, PRIVATE);// publish temperature in Celsius
        if (t > 40 && t < 75) Particle.publish("Temperature", "High", 60, PRIVATE);//notify in case of emergency
        qsort(humis, 21, sizeof(float), compare);//sort to put any outliers on either end of the array
        float h = humis[11];//median value in the array
        sprintf(humi, "%.1f", h);//convert float to a string with 2 digitis after decimal point
        if (h < (humis[11] + 1) && h > (humis[11] - 1)) Particle.publish("Humidity / %", humi, 60, PRIVATE);//publish humidity in %
        int spillage = analogRead(spillsensor);//Spillage sensor
        float spillage_a = spillage;
        if (isnan(spillage_a)) {
            spillage = 4095;
        }
        float spillage_b = (4100 - spillage);
        float spillage_c = ((spillage_b)/37);
        sprintf(spill,"%.2f", spillage_c);
        Particle.publish("Spill", spill, 60, PRIVATE);
        if (spillage_c > 10) {
            Particle.publish("Spill", "detected", 60, PRIVATE);
            digitalWrite(pst, LOW);//power switch off if spillage detected
        }
        if (t > 40) {
              digitalWrite(pst, LOW);//power switch off if high temperature recorded
             }
             if (digitalRead(pst)==LOW) {
            Particle.publish("Power Switch", "off", 60, PRIVATE);//publish to indicate power switch is off
        }
      }
  }
}

int compare (const void * a, const void * b) {//qsort and median filter codes
    float fa = *(const float*) a;
```

```
    float fb = *(const float*) b;
    return (fa > fb) - (fa < fb);
}

int switchToggle(String command) {//remote control for power switch
    if (command=="on") {
        digitalWrite(pst,HIGH);
        return 1;
    }
    else if (command=="off") {
        digitalWrite(pst,LOW);
        return 0;
    }
    else {
        return -1;
    }
}//end of code
```

ADDITIONAL FIGURES

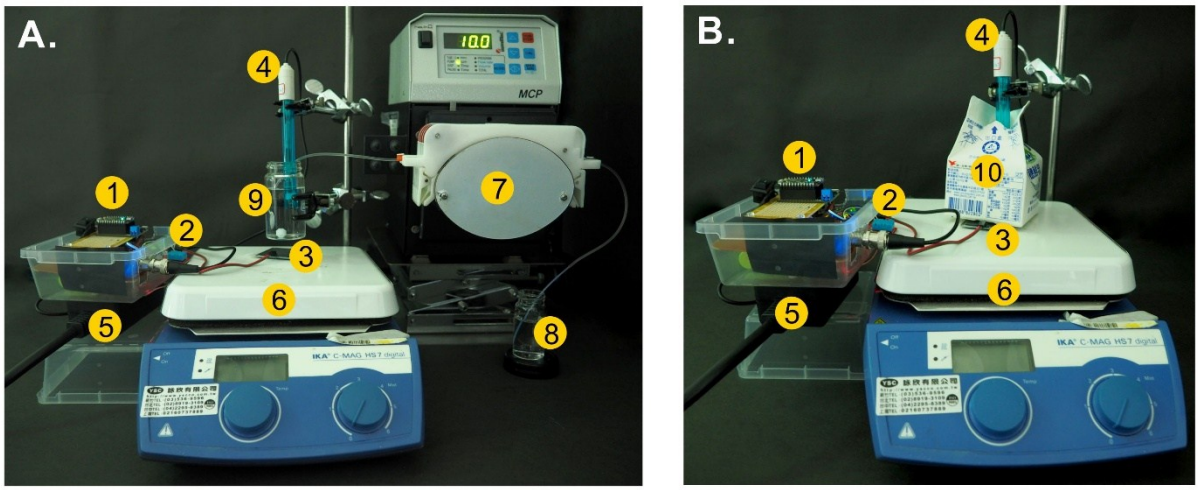

Figure S1. Photographs of the experimental system: (A) monitoring pH oscillations; (B) monitoring pH of milk undergoing microbial fermentation. 1. *Particle Photon* Wi-Fi module; 2. Temperature and humidity sensor; 3. Spillage sensor; 4. pH electrode; 5. Power switch tail; 6. Magnetic stirrer; 7. Peristaltic pump; 8. Reactant 1; 9. Reactant 2; 10. Milk pack.

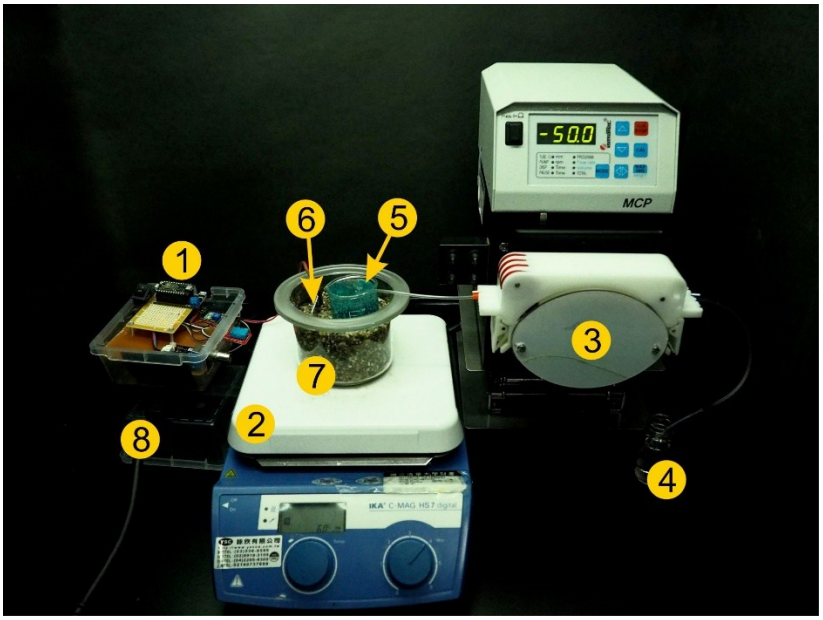

Figure S2. Photograph of a typical experimental system involving heating (*cf.* **Movie S1**): 1. *Particle Photon* Wi-Fi module; 2. Hot plate/magnetic stirrer; 3. Peristaltic pump; 4. Water reservoir; 5. Reaction vessel (water with methylene blue dye); 6. Spillage sensor; 7. Sand bath; 8. Power switch tail.

ADDITIONAL MOVIE

Movie S1 (separate file). Video demonstrating auto-shutdown of hot plate/magnetic stirrer and peristaltic pump upon detection of reactant spillage (*cf.* **Figure S2**). Reaction vessel containing methylene blue dyed water was heated (60 °C) in a sand bath. The spillage sensor was intentionally kept a little above the sand surface to clearly show the raise in spilled solution. Extra water was poured to quickly raise the level of spilled solution in the sand bath (to mimic an overflow event). The level of immersion of spillage sensor may be further lowered to the bottom of the sand bath to shorten the response time.