Ver.1.0 2019-03-12

K4 スマートIoTシステム ビジネス入門

(制作回答例)

2 - プロトタイピング

大黒　篤

(MCPC)

WASEDA NEO Smart SE

5-4

Raspberry Piによる プロトタイピング

＜学習目標＞

- Raspberry Piを使った作品の制作
- 作品の発表
- 振り返り

3.3 Raspberry Piによるプロトタイピング実践（制作）

- 制作課題
  a. 課題１（ラーメンタイマー）

     ボリュームで時間を秒単位で設定し、タクトスイッチでタイマー開始。タイムアップとなったら、LEDとビープで知らせる

  b. 課題２（リモート環境取得）

     メールを特定のアドレスに送ると、その時の温度、湿度を計測し、不快指数を計算してから、返信でそれらの情報を返す。

  c. 課題３（お天気目覚まし時計）

     あらかじめ設定しておいた時刻や地域情報をもとに、アラーム時刻となったら、LEDを点滅＆ビープを鳴らし、当日の天気予報を音声またはLCDで教えてくれる。

- 上記から2つを選択し、14:15までに検討・制作する。
- 制作した課題は、各チームで発表資料を準備してもらい、15:00から発表してもらう。

制作例　課題１（ラーメンタイマー）

- 仕様
  - ボリュームを回して、セットしたい時間（1〜10分の間で調整可能）をLCDに表示させる。
  - その状態でタクトスイッチを押すことで、指定した時間（n分）でタイマーがスタートする。
  - LCD上には、カウントダウンのタイマー（nから始まり、n-1、n-2...1, 0）が表示される。
  - 以下のように動作する：
    - n〜2分の間は、2秒間隔（1秒点灯、1秒消灯）でLEDが点滅する。
    - 2〜1分の間は、1秒間隔（0.5秒点灯、0.5秒消灯）でLEDが点滅する。
    - 0分となったら、0.4秒間隔（0.2秒点灯、0.2秒消灯）でLEDが点滅し、同間隔でビープが鳴る
    - 30秒が経過するか、またはタクトスイッチが押されたらLEDは消灯し、ビープも止まる。
  - カウントダウンの時に、タクトスイッチを押すと、タイマーはキャンセルされる。
- 配線
  - LEDとタクトスイッチの回路は、p.34の「プロトタイピング実践（３）」と同じ。
  - ビープの回路は、p.31の「プロトタイピング実践（４）」で、16番ピン(GPIO23)を17番ピン(GPIO22)に変更する。
  - ボリュームの回路は、p.39の「プロトタイピング実践（６）」と同じ。
  - LCDの回路は、p.46の「プロトタイピング実践（８）」と同じ。
- プログラム
  - できるだけシンプルに、難しい機能は使わずに実装します。コードを、以下に示します。

talk_weather.py

制作例　課題１（ラーメンタイマー）

ramen_timer.py

1/4

```
#coding: utf-8
#  Ramen timer (KADAI-1)

import wiringpi as pi
import time, sys
import mcp3008
import i2c_lcd

# LCDに指定された2行を表示する
def displayLCD(line1, line2 = ""):
  i2c_lcd.lcd_init()       # LCDをクリアする
  i2c_lcd.lcd_string(line1, i2c_lcd.LCD_LINE_1)
  i2c_lcd.lcd_string(line2, i2c_lcd.LCD_LINE_2)


# メイン処理
BUZZER_PIN = 22
LED_PIN = 23
SW_PIN = 24
CH = 0

pi.wiringPiSetupGpio()
i2c_lcd.lcd_init()          # i2c_lcdを初期化する

pi.pinMode(BUZZER_PIN, pi.OUTPUT)
pi.pinMode(LED_PIN, pi.OUTPUT)
pi.pinMode(SW_PIN, pi.INPUT)
pi.pullUpDnControl(SW_PIN, pi.PUD_UP)
```

2/4

```
try:
  while True:

    # タイマー設定処理
    if (stop):
      pi.digitalWrite(LED_PIN, pi.LOW)
      pi.digitalWrite(BEEP_PIN, pi.LOW)

    setTme = 0   # タイマ設定時間(1..10分)
    buf = ""
    while (pi.digitalRead(SW_PIN) == pi.HIGH):
      data = mcp3008.readAdcValue(CH)
      setTime = int(data / 110) + 1    # 1 .. 10に変換する
      buf = "Time: {:2}min".format(setTime)
      displayLCD(buf)
      time.sleep(0.05)

    # ボタンが押されたので、タイマーを開始する
    startTime = time.time()     # タイマーの開始時間
    endTime = startTime + (setTime * 60)
                                  # タイマーの終了時間
    stop = False
    displayLCD(buf, "START..")
```

制作例　課題１（ラーメンタイマー）

3/4

```
# n..2分の時
t = startTime
while (t < endTime - 60.0):
  if (pi.digitalRead(SW_PIN) == pi.LOW):
    stop = True    # SWが押されたらキャンセルする
    break
  pi.digitalWrite(LED_PIN, pi.HIGH)
  time.sleep(1.0)
  pi.digitalWrite(LED_PIN, pi.LOW)
  time.sleep(1.0)
  t = t + 2.0
if (stop):
  continue      # タイマー設定処理に戻る

# 2..1分の時
while (t < endTime):
  if (pi.digitalRead(SW_PIN) == pi.LOW):
    stop = True    # SWが押されたらキャンセルする
    break
  pi.digitalWrite(LED_PIN, pi.HIGH)
  time.sleep(0.5)
  pi.digitalWrite(LED_PIN, pi.LOW)
  time.sleep(0.5)
  t = t + 1.0
if (stop):
  continue      # タイマー設定処理に戻る
```

4/4

```
  # 0分の時(タイムアウトした時)
  displayLCD("TIME OUT!!")
  while (t < endTime + 30):
    if (pi.digitalRead(SW_PIN) == pi.LOW):
      break        # SWが押されたらキャンセルする
    pi.digitalWrite(LED_PIN, pi.HIGH)
    pi.digitalWrite(BUZZER_PIN, pi.HIGH)
    time.sleep(0.2)
    pi.digitalWrite(LED_PIN, pi.LOW)
    pi.digitalWrite(BUZZER_PIN, pi.LOW)
    time.sleep(0.2)
    t = t + 0.4

except KeyboardInterrupt:
  pi.digitalWrite(LED_PIN, pi.LOW)
  i2c_lcd.lcd_init()       # LCDをクリアする
  sys.exit(0)
```

うまく動かすコツは、
部分、部分を一つずつ作り、
作った部分の動作を確認しながら、
少しずつ機能を足していくこと。

参考（制作：お天気目覚まし時計）

```
#! /usr/bin/env python3
# coding: utf-8

import json, sys, os
import urllib.request, urllib.parse

def get_forecast(city):
    jsonUrl = http://weather.livedoor.com/forecast/webservice/json/v1?
    url = jsonUrl + "city=" + city
    site = urllib.request.urlopen(url)
    jsonData = json.loads(site.read().decode("utf-8"))
    return jsonData

if __name__ == "__main__":
    kanazawa_city = "170010" # 石川県金沢市
    data = get_forecast(kanazawa_city)

    try:
        weather = data["location"]["city"] + "地方、明日の天気は" + data["forecasts"][1]["telop"] + "でしょう。"
        weather = weather.replace("曇", "曇り")
        tempMax = data["forecasts"][1]["temperature"]["max"]["celsius"]
        weather += "最高気温は、" + tempMax + "度。"
        tempMin = data["forecasts"][1]["temperature"]["min"]["celsius"]
        tempMin = tempMin.replace("-", "マイナス")
        weather += "最低気温は、" + tempMin + "度でしょう。"

    except TypeError:
        pass

    print(weather)    cmd = "/home/pi/aquestalkpi/AquesTalkPi " + weather + " -s 90 | aplay -q"
    os.popen(cmd).readline().strip()
```

talk_weather.py