New Jersey Institute of Technology
Department of Engineering Technology
Spring 2017

Smart Climate Control
Technical Manual

1 May 2017

Siddharth Patil
Jeffrey St. Jean
Joseph Skwiat
ECET 402 — Professor Alex Blinder

©2017 Siddharth Patil, Jeffrey St. Jean, Joseph Skwiat.
All rights reserved, unless otherwise specified.

Unless otherwise indicated, to the extent possible, the copyright holders hereby offer the information and materials in this document up to but not including appendix E ("Document") as-is and as-available, and make no representations or warranties of any kind concerning the information in this document, whether express, implied, statutory, or other. This includes, without limitation, warranties of title, merchantability, fitness for a particular purpose, non-infringement, absence of latent or other defects, accuracy, or the presence or absence of errors, whether or not known or discoverable. In no event shall the copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the Document or the use or other dealings in the information in the Document.

The information and materials in appendix E are © of their respective copyright holders.

Produced with LaTeX 2_ε.

Contents

I Introduction **1**

1 Project description **1**

2 Acknowledgements **1**

II Design **3**

3 Block diagrams **3**

3.1 First-level block diagram 3

3.2 Second-level block diagram 4

4 Hardware **6**

4.1 Microcontroller 6

4.2 Power 6

4.3 Operation 7

4.3.1 Pulse reading 7

4.3.2 Signal conditioning 7

4.3.3 Pulse counting 8

4.4 Enclosure 8

4.5 Construction 8

5 Software **9**

5.1 Programming language 9

5.2 Major modules 9

5.2.1 Network configuration 9

5.2.2 Web interface 10

5.2.3 Smart outlet control 10

III Appendices **12**

A Deliverables **13**

A.1 Summary of Deliverables 13

A.2 Deliverable #1: Smart band's WiFi connectivity 14

A.2.1 Deliverable #1 Test Procedures 15

A.3 Deliverable #2: Smart outlet configuration and control 16
A.3.1 Deliverable #2 Test Procedures 17
A.4 Deliverable #3: Pulse sensor integration 18
A.4.1 Deliverable #3 Test Procedures 19
A.5 Deliverable #4: Smart band prototype 20
A.5.1 Deliverable #4 Test Procedures 21

B Source code **22**
B.1 Network configuration 22
B.2 Web interface 22
B.3 Pulse detection and smart outlet switching 24
B.4 Smart outlet control commands 26

C Bill of materials **27**

D Software licensing **28**

E Data sheets **30**

List of Figures

1 First-level block diagram 3
2 Second-level block diagram 4
3 Circuit diagram of smart band 6
4 Input and output waveforms of signal conditioning circuit . . . 7
5 Flowchart of smart band software 9
6 Schematic of smart band 14
7 Simplified schematic of smart outlet 16
8 Design of smart band 20

Listings

1 `init.lua` 22
2 `web.lua` 22
3 `index.html` 24
4 `error.html` 24
5 `control.lua` 24
6 `Smart_outlet_on.dat` (hex dump) 26
7 `Smart_outlet_off.dat` (hex dump) 26

I. Introduction

1 Project description

The problem

Traditional climate control systems do not offer facility for direct control based on a user or occupant's physiological state. This functionality can be useful in driving climate control systems toward thermal comfort of its users.

One such example involves women suffering from menopausal hot flashes. Up to 75% of perimenopausal women suffer from hot flashes in the US.[1] Hot flashes are accompanied by an increase in heart rate, which can be measured and analyzed as an indicator or predictor.

Our solution

This project is a system for driving $120\,V_{ac}$ thermal comfort devices based on the user's pulse rate. Pulse rate differs from heart rate in that it is a function of heart rate. Pulse rate is closely related to heart rate, enough so that measurement of pulse rate serves as an acceptable substitute for measurement of heart rate.[2]

Our smart climate control system consists of three parts: a smart band, a smart outlet, and a thermal comfort device. The smart band is essentially an activity tracker, capturing the pulse rate information of its wearer and processing it. The smart outlet allows control of a $120\,V_{ac}$ device through the network; the user would connect a thermal comfort device such as a fan or air conditioner to the smart outlet.

The primary feature of our smart climate control system is, for the most part, no need for manual control. Once a threshold pulse rate has been set by the user, the smart climate control system will automatically drive the thermal comfort device.

2 Acknowledgements

We would like to thank Dylan Davies and Professor Cong Wang of the NJIT Controls, Automation, and Robotics Laboratory (CAR Lab) for resources used during development and testing, as well as technical guidance during the development process.

[1] North American Menopause Society. *Hot Flashes*. URL: https://www.menopause.org/for-women/sexual-health-menopause-online/causes-of-sexual-problems/hot-flashes (visited on 05/01/2017).

[2] Hangsik Shin. "Ambient temperature effect on pulse rate variability as an alternative to heart rate variability in young adult". In: *Journal of Clinical Monitoring and Computing* 30.6 (Dec. 1, 2016), pp. 939–948.

We would like to thank Christopher Karanievski and Professor James Finne of Middlesex County College in Edison, NJ for resources used during development, as well as technical guidance during the development process.

II. Design

3 Block diagrams

3.1 First-level block diagram

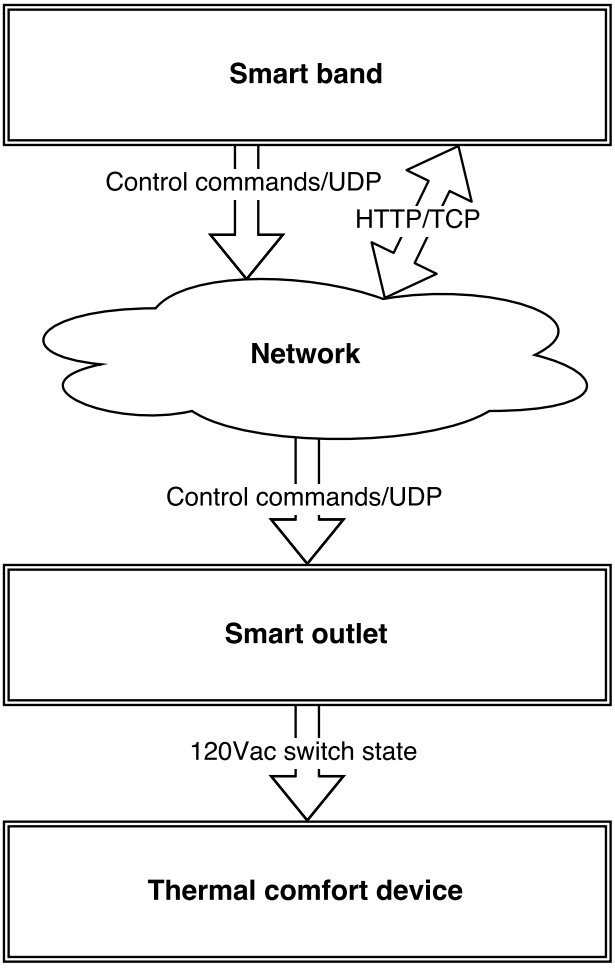


Figure 1 First-level block diagram

The smart band interacts with the smart outlet through simplex (one-way) network communication, in the form of control commands encapsulated by UDP packets. The smart band is configured through an HTTP interface carried by a TCP connection. Once a control command is received the smart outlet, the outlet switches $120\,V_{ac}$ to the thermal comfort device, turning it on or off depending on the control command received.

3.2 Second-level block diagram

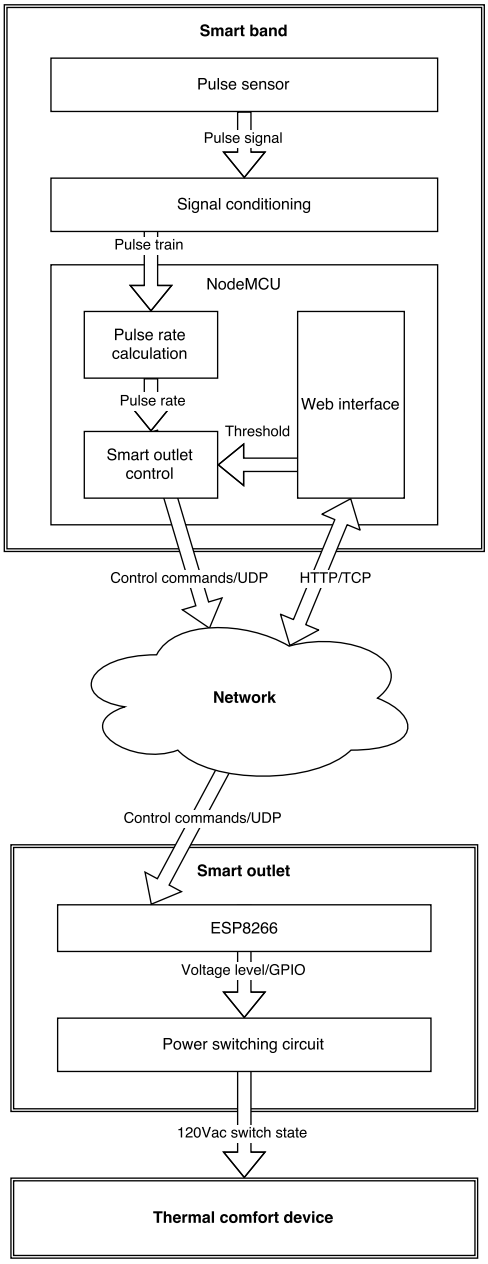


Figure 2 Second-level block diagram

Within the smart band, the pulse sensor produces a signal which is passed through signal conditioning. The signal conditioning produces a pulse train based on the pulse signal, which is then passed to the NodeMCU. The NodeMCU performs a pulse rate calculation using the pulse train, and passes the calculated pulse rate to the smart outlet controller. The Web interface, accessible via HTTP over a TCP connection, passes a user-specified threshold to the smart outlet controller. The smart outlet controller issues control commands to the smart

outlet based on a comparison between the pulse rate and the threshold. The control commands are carried in UDP datagrams.

Once the ESP8266 microcontroller onboard the smart outlet receives and processes the control commands, it drives the power switching circuit. The power switching circuit switches $120\,V_{ac}$ to the thermal comfort device, turning it on or off depending on which control command was received.

4 Hardware

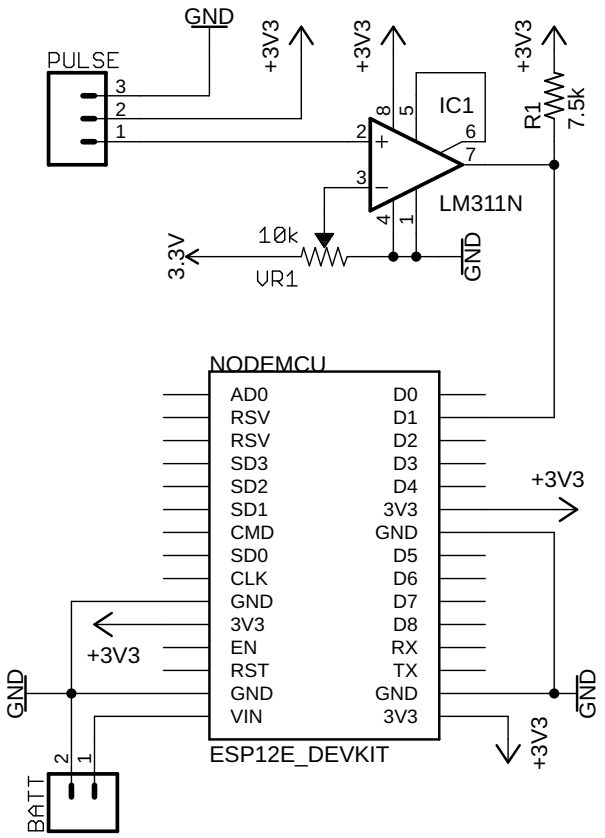


Figure 3 Circuit diagram of smart band

4.1 Microcontroller

The microcontroller platform chosen for the smart band was the NodeMCU v2 module, denoted NODEMCU. This module is driven by an ESP8266 microcontroller by Espressif Systems. This module was chosen for its low cost, integrated WiFi connectivity, and development toolchain.

The smart outlet uses an ESP8266 microcontroller by Espressif Systems.

4.2 Power

The smart band can be powered either with a $9\,V_{dc}$ battery via clip-on terminals, or with a $5\,V_{dc}$ USB power supply (power bank, USB cell phone charger, etc.) via Micro-USB connector onboard NODEMCU. An AMS1117 low-dropout linear voltage regulator onboard NODEMCU provides $3.3\,V_{dc}$ to the ESP8266 and other internal circuitry. This $3.3\,V_{dc}$ supply is also available on three of NODEMCU's pins, denoted +3V3.

The smart outlet is powered by $120\,V_{ac}$ line power, via its NEMA 5-15 connector.

4.3 Operation

The smart band operates in three stages: (1) pulse reading, (2) signal conditioning, and (3) pulse counting.

4.3.1 Pulse reading

The pulse sensor module, denoted PULSE, is powered by +3V3. This module is placed on the user's wrist, fingertip, or earlobe. The module generates a variable waveform representing the pulse rate of the user, known as the pulse signal.

4.3.2 Signal conditioning

The pulse signal propagates from PULSE output pin 1 to noninverting input pin 3 of an LM311 differential voltage comparator, denoted IC1. IC1, powered by +3V3, is configured to operate as a noninverting voltage-level comparator, with the reference voltage level set from $0\,V_{dc}$ to $3.3\,V_{dc}$ through a $10\,k\Omega$ potentiometer configured as a voltage divider, denoted VR1. IC1 features an open-collector output, so output pin 7 is pulled to 3.3 V through a $7.5\,k\Omega$ current-limiting resistor, denoted R1.

This stage generates a rectangular pulse train from the pulse signal. This stage can be mathematically modeled by equation (1), where V_{ref} is the reference voltage provided by VR1.

$$\text{Output} = \begin{cases} 3.3\,V & \text{Input} > V_{ref} \\ 0\,V & \text{Input} \leq V_{ref} \end{cases} \tag{1}$$

The fast and sharp edges of the pulse train make it suitable for triggering edge-driven interrupts in microcontrollers. This pulse train can be seen compared to the original variable-waveform pulse signal in figure 4.

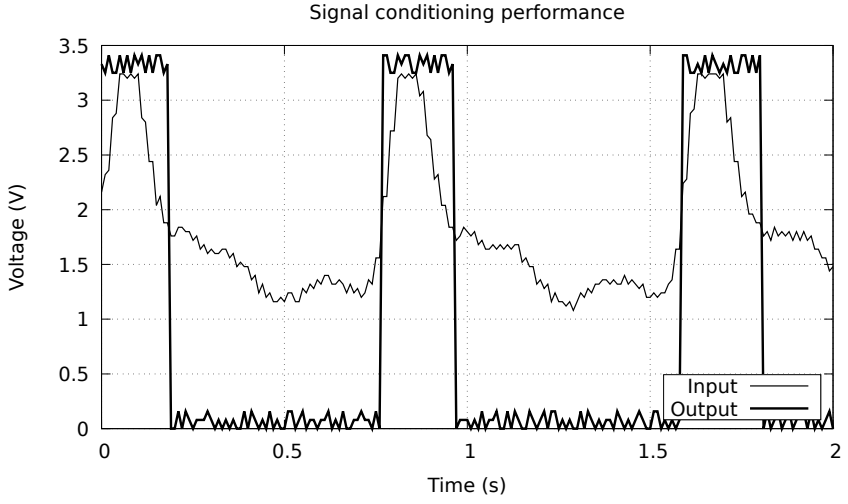


Figure 4 Input pulse signal and output pulse train of signal conditioning circuit. Pulse train changes level when input crosses approximately $1.88\,V_{dc}$.

4.3.3 Pulse counting

The pulse train propagates from IC1 output pin 7 to NODEMCU GPIO pin D1. D1 is configured as an input pin operating in interrupt mode. The software program running on NODEMCU operates on a falling-edge interrupt triggered every time the voltage level on D1 transitions from a high level to a low level (in this case, from 3.3 V to 0 V.)

4.4 Enclosure

The enclosure for the smart band measures 70 mm by 50 mm by 40 mm, with a wall thickness of 2 mm. A 10 mm by 12 mm opening was included on one side of the enclosure for external access to the Micro-USB port on the NodeMCU, and a 7 mm by 10 mm opening was drilled below it for a rocker-style power switch. A circular opening 5 mm in diameter was drilled on the opposite side to allow the pulse sensor to extend outside the enclosure.

The enclosure was produced with polyethylene terephthalate glycol (PETG) filament using a fused deposition modeling (FDM) 3D printer.

4.5 Construction

The circuitry for the smart band was mounted and wired on a copper-clad perfboard measuring 60 mm by 40 mm, itself mounted inside the enclosure. The board features sockets for the NodeMCU and LM311 IC, and Phoenix connectors for the pulse sensor and 9 V_{dc} battery clip.

5 Software

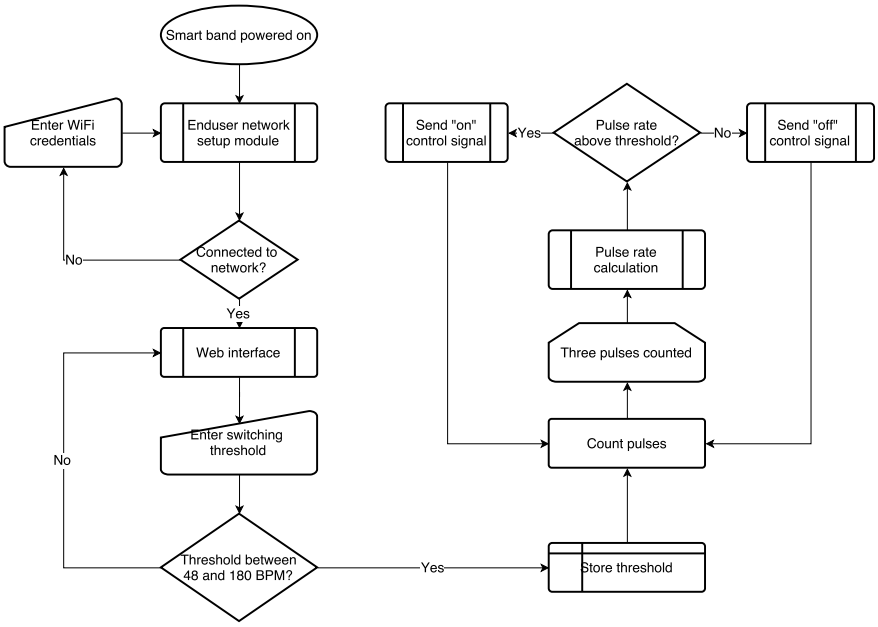


Figure 5 Flowchart of smart band software

The NodeMCU uses a third-party Free/open source firmware package consisting of a Lua interpreter and modules offering high-level functionality through Lua. The firmware package itself is written in C. Modifications were made to the firmware package in order to make the Web interface accessible to the end user.

5.1 Programming language

The program developed for the smart band is written in Lua, accompanied by HTML source files for the Web interface. Development for the NodeMCU in Lua is primarily event-driven and asynchronous; much of the smart band's functionality is implemented through software interrupts, with use of timing routines and loops kept to a minimum.

5.2 Major modules

5.2.1 Network configuration

Upon receiving power, the NodeMCU runs an enduser network setup module (found in listing 1) which allows the user to connect the microcontroller to his/her own WiFi network. Upon running the enduser network setup module, the microcontroller creates its own WiFi access point which the user connects to. Once the user has connected to the microcontroller's access point, a captive portal intercepts the user's HTTP (Web browser) requests and redirects him/her to the setup interface. The user chooses his/her WiFi access point from a list and provides the password or authentication key. Once he/she submits this information, the module attempts to connect to the desired network. If the connection is successful, the module displays the IP address assigned by the

desired network, as well as the port number the smart band Web interface will be running on. The module then shuts itself down.

Modifications to the enduser network setup module were necessary in order for the module to display the received IP address.and port number of the microcontroller. These modifications are not included in this manual; they can be found at `https://github.com/spatil47/nodemcu-firmware/commits/master`.

5.2.2 Web interface

The Web interface is implemented through a small HTTP server capable of handling requests made with the HTTP `GET` request method, and is accessed through the currently-connected WiFi network at a dynamically-assigned IP address and a fixed port number of 8000, both displayed to the user by the network configuration module.

The interface consists of a set of HTML pages, some stored in flash memory and some stored in the program itself. The interface contains an HTML form (found in listing 3) which offers the user the opportunity to set his/her desired switching threshold in BPM, used by the smart outlet control as one of the switching conditions. The user is only allowed to enter a value between 48 BPM to 180 BPM (the same range used in one of the pulse rate filtering algorithms); the interface will serve an error message (found in listing 4) if the user enters a value outside this accepted range, at which point he/she will be offered another chance to enter an accepted value.

When the user submits an accepted value, the HTTP server parses the contents of the `GET` request, saving the accepted value to a file. The HTTP server then serves a success message (found in listing 2) and shuts itself down. The saved file containing the accepted value is accessed by the smart outlet control module, which uses it as a switching condition.

5.2.3 Smart outlet control

The smart outlet control module (found in listing 5) accepts a pulse train on a GPIO pin from the pulse sensor conditioning circuit. It counts the number of pulses through a falling-edge interrupt, and calculates the pulse rate in BPM by continuously finding the time difference between the first and last pulse for every three pulses, calculating the ratio of three pulses to the time difference, and multiplying the ratio by a constant in order to convert it to units of BPM.

The output series of this calculation contains high-frequency noise, so several filters are used to denoise this data. The first is a type of soft limiting filter which checks if the latest pulse rate is within an acceptable range of 48 BPM to 180 BPM. If it is within the range, the value is passed to the second filter. If it falls outside the range, no new value is passed to the second filter, which is not activated. This filter can be mathematically modeled by equation (2), where S_n is the current output of the soft limiting filter, R_n is the current output of the pulse rate calculation, and S_{n-1} is the last output of the soft limiting filter.

$$S_n = \begin{cases} R_n & 48 \leq R_n \leq 180 \\ S_{n-1} & 48 \nleq R_n \nleq 180 \end{cases} \tag{2}$$

The second is a simple moving-average filter, which calculates the average of the current and last pulse rates. The output of these two filters is then passed to

the switching portion of the control program. This filter can be mathematically modeled by equation (3), where A_n is the current output of the simple moving-average filter, and A_{n-1} is the last output of the simple moving-average filter.

$$A_n = \frac{S_n + A_{n-1}}{2} \tag{3}$$

The switching controller is a bang-bang controller with zero hysteresis. If the value passed to it is less than the switching threshold set by the user, it will transmit the "off" command in a UDP packet to the smart outlet (found in listing 7). If the value is higher than the switching threshold, it will transmit the "on" command in a UDP packet to the smart outlet (found in listing 6). This controller can be mathematically modeled by equation (4), where T is the user-configured switching threshold.

$$\text{Smart outlet state} = \begin{cases} \text{On} & A_n > T \\ \text{Off} & A_n \leq T \end{cases} \tag{4}$$

The smart outlet is controlled through a proprietary command set over UDP. Packets containing the necessary commands (found in listings 6 and 7) were captured with Wireshark and saved onboard the NodeMCU in its flash storage. The appropriate command is transmitted when the switching condition is met each calculation cycle.

III. Appendices

See next pages for deliverables, project source code, bill of materials, software licensing, and data sheets.

A Deliverables

A.1 Summary of Deliverables

Deliverable	Description	Steps in Testing	Due Date
Smart band's WiFi connectivity	Smart band can connect to WiFi access point and host Web interface.	9	20 Feb
Smart outlet configuration and control	Smart band can connect to smart outlet and send control commands.	22	6 Mar
Pulse sensor integration	Smart band can receive and interpret pulse sensor data.	23	3 Apr
Smart band prototype	Smart band can control smart outlet based on pulse sensor data.	27	17 Apr

A.2 Deliverable #1: Smart band's WiFi connectivity

In Deliverable #1, we will be demonstrating the NodeMCU, which is the microcontroller module powering the smart band. The NodeMCU will drive the smart outlet without the need for manual control. We will be demonstrating the smart band's ability to connect to a WiFi network and host a Web interface. The Web interface will be the control panel for the user, where he/she will enter information required for the smart band's autonomous operation.

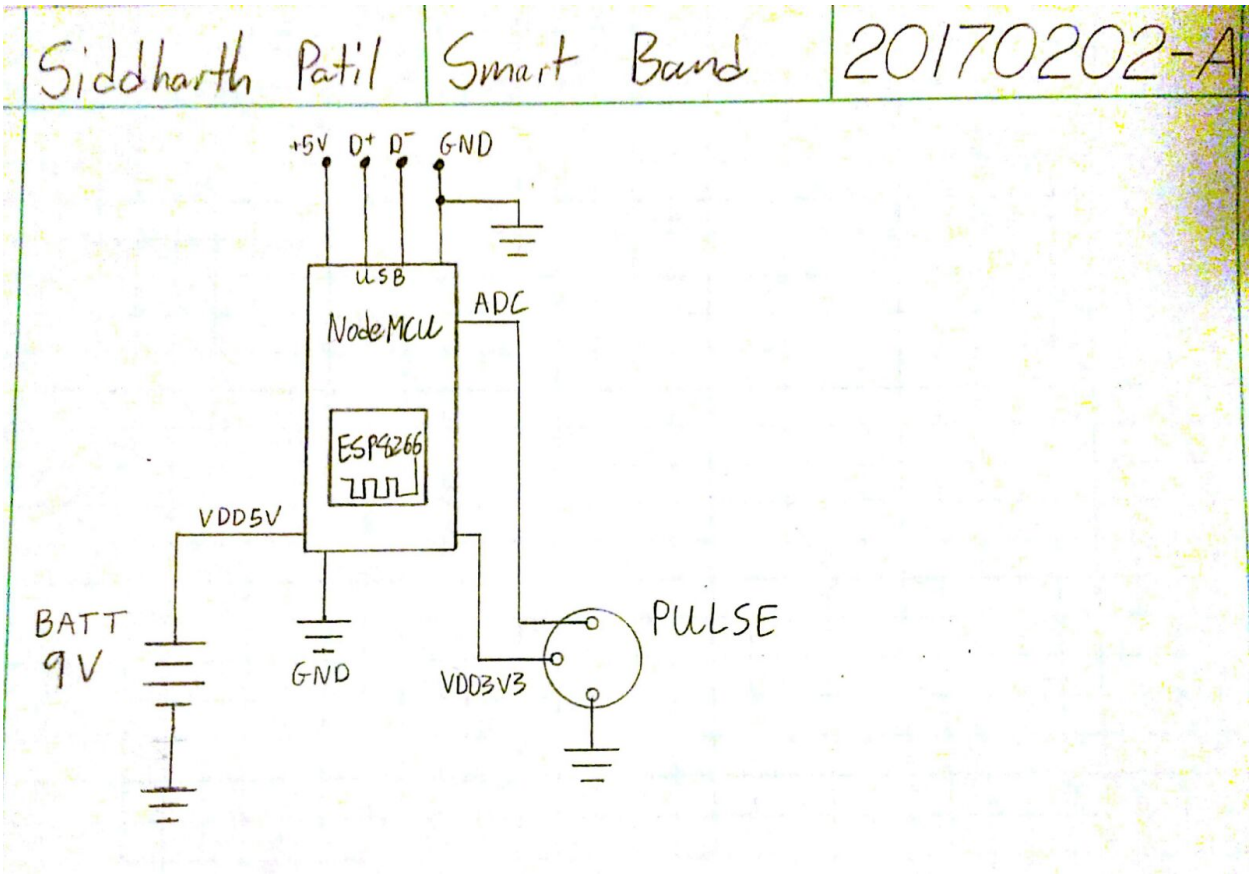


Figure 6 Schematic of smart band

Figure 3 shows the major signals and power connections in the smart band. BATT is a $9\,V_{dc}$ battery, connected to VDD5V.[3] PULSE is the pulse sensor, accepting $3.3\,V_{dc}$ from pin VDD3V3, and sending an analog signal to pin ADC. The four pins at the top of the NodeMCU are used for USB connections, which in this case would be used for receiving debugging information via UART.

The due date for this deliverable is February 20, 2017.

[3] The pin name VDD5V is a misnomer, as the regulator connected to this pin has a maximum input voltage of $20\,V_{dc}$.

A.2.1 Deliverable #1 Test Procedures

Follow the test procedure steps below and record results in Deliverable #1 Test Report Form.

Connecting smart band to WiFi access point

A.2.1.1.1 Connect 9 V battery to clip-on terminals, and watch LEDs on NodeMCU. If they blink or flash, proceed to step A.2.1.1.2.

A.2.1.1.2 Press RST button on NodeMCU module. If LEDs blink or flash in the same pattern as they did in step A.2.1.1.1, proceed to step A.2.1.1.3.

A.2.1.1.3 Wait 10 seconds. Proceed to step A.2.1.1.4.

A.2.1.1.4 Using WiFi-enabled device, search for and connect to network with SSID starting with the prefix "SetupGadget_". Proceed to step A.2.1.1.5.

A.2.1.1.5 Open browser on WiFi-enabled device and browse to `http://example.com`. If you see a login page with fields for WiFi SSID and password entry, proceed to step A.2.1.1.6.

A.2.1.1.6 Enter SSID and password of commonly-used WiFi access point, and wait for success or failure message. If successful, proceed to step A.2.1.1.7.

A.2.1.1.7 Success message shows an IP address. Write it down and click its link. Proceed to step A.2.1.2.1 of test procedure set "Navigating smart band's Web interface" in appendix A.2.1.

Navigating smart band's Web interface

A.2.1.2.1 If IP address link was not clicked in step A.2.1.1.7, navigate to IP address. If Web interface displays, proceed to step A.2.1.2.2.

A.2.1.2.2 Web interface contains a single button element, labeled "IP address". Click this button. If IP address displayed matches IP address in address bar of web browser, test was successful.

A.3 Deliverable #2: Smart outlet configuration and control

In Deliverable #2, we will demonstrate the NodeMCU's ability to drive the smart outlet. The NodeMCU will send control commands to the smart outlet to switch it on or off. This is integral to the operation of the smart band, as it is how the band will control the thermal comfort device.

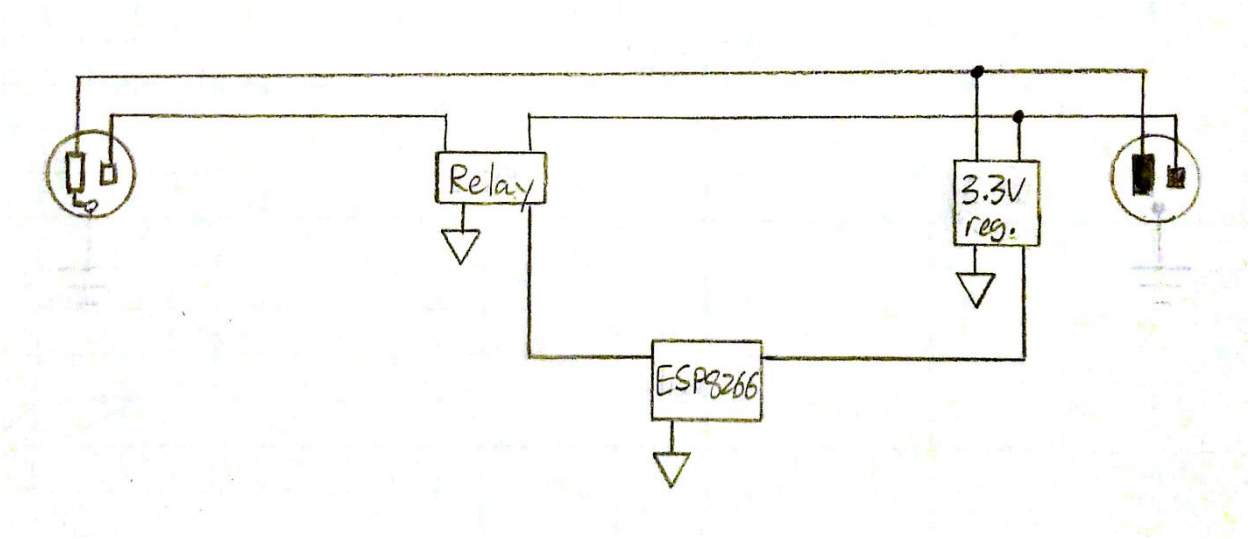


Figure 7 Simplified schematic of smart outlet

Figure 7 shows a highly-simplified schematic of the smart outlet, designed to emphasize the power distribution and relay control signals. All smart outlets are designed on the same basic principle, with the primary differences being specific hardware and software implementation.

The due date for this deliverable is March 6, 2017.

A.3.1 Deliverable #2 Test Procedures

Follow the test procedure steps below and record results in Deliverable #2 Test Report Form.

Connecting smart band and smart outlet to WiFi access point

A.3.1.1.1 Follow procedure set “Connecting smart band to WiFi access point” in appendix A.2.1. This procedure set consists of 7 steps. Proceed to step A.3.1.1.2.

A.3.1.1.2 Follow procedure set “CT-065W Standard Quick Guide”, subsection “Smartphone Setup”, attached to this document. This procedure set consists of 11 steps. Proceed to step A.3.1.1.3.

A.3.1.1.3 Use smartphone app to switch smart outlet on and off. Proceed to step A.3.1.1.4.

A.3.1.1.4 Follow procedure set “Navigating smart band’s Web interface”, found in appendix A.2.1. This procedure set consists of 2 steps. Proceed to step A.3.1.1.1 of test procedure set “Sending a control command from smart band to smart outlet” in appendix A.3.1.

Sending a control command from smart band to smart outlet

A.3.1.1.1 Web interface contains two button elements. One of these buttons is labeled “On”, and the other button is labeled “Off”. Click each button and observe the smart outlet’s switching state. Relay should switch on when “On” button is clicked, and relay should switch off when “Off” button is clicked. If both these conditions are met, test procedure is successful.

A.4 Deliverable #3: Pulse sensor integration

In Deliverable #3, we will demonstrate the smart band reading and interpreting data from the pulse sensor. The pulse sensor data will be used by the smart band to make control decisions. In this deliverable, the user's heart rate will be displayed in the Web interface. The tester will confirm whether this pulse rate matches an independently-measured pulse rate using a manual measurement technique or an electronic monitor.

Schematic of pulse sensor located in appendix E.

The due date for this deliverable is April 3, 2017.

A.4.1 Deliverable #3 Test Procedures

Follow the test procedure steps below and record results in Deliverable #3 Test Report Form.

Reading heart rate from pulse sensor

A.4.1.1 Follow procedure set “Connecting smart band and smart outlet to WiFi access point”, found in appendix A.3.1. (Step A.3.1.1.3 may be skipped.) This procedure set consists of 21 steps. Proceed to step A.4.1.2.

A.4.1.2 Affix the pulse sensor to the wrist, fingertip, or earlobe. Wrist is highly preferred. Proceed to step A.4.1.3.

A.4.1.3 Web interface contains one button element, labeled “Pulse Rate”. Click this button and wait for 6 seconds. A number should appear on the page. This number is the tester’s current pulse rate, measured in beats per minute. Verify this number using manual measurement techniques or an electronic heart rate monitor. If the number on the page matches the heart rate value produced by the verification procedure or device, test procedure is successful.

A.5 Deliverable #4: Smart band prototype

In Deliverable #4, we will be demonstrating the smart band's ability to capture data from the pulse sensor and control the smart outlet. The user will set a target heart rate through the Web interface, which the NodeMCU will use to compare the pulse sensor data and make a switching decision.

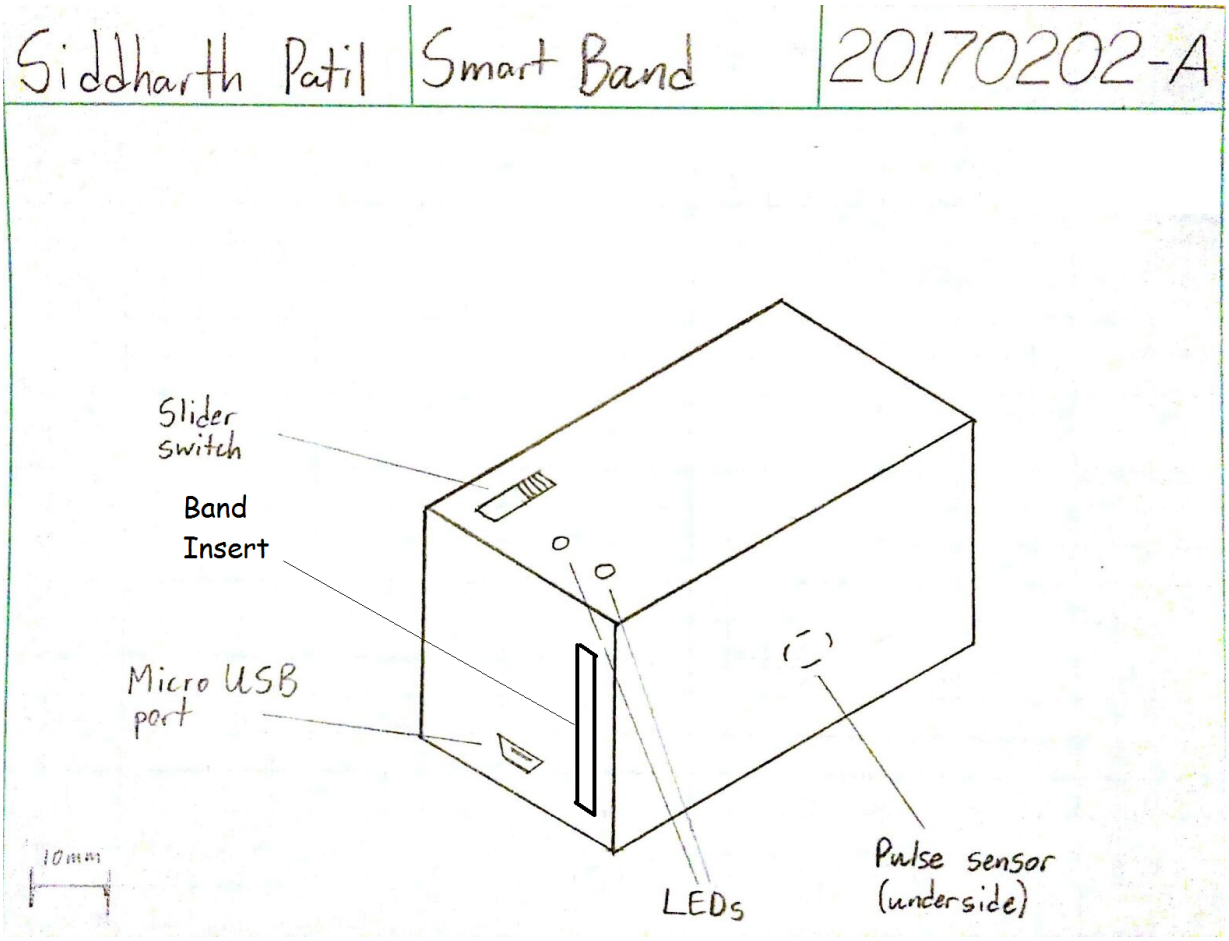


Figure 8 Design of smart band

The due date for this deliverable is April 17, 2017.

A.5.1 Deliverable #4 Test Procedures

Follow the test procedure steps in appendices A.5.1 to A.5.1 and record results in Deliverable #4 Test Report Form.

Configuring smart band for target heart rate

A.5.1.1.1 Follow steps A.4.1.1 and A.4.1.2 of procedures found in appendix A.4.1. This procedure subset consists of 22 steps. Proceed to step A.5.1.1.2.

A.5.1.1.2 Web interface contains an input field labeled “Target pulse rate”, and a button labeled “Set”. Enter the pulse rate you wish the smart outlet to switch at, and click the button. Proceed to step A.5.1.2.1 of test procedure set “Switching on smart outlet based on pulse rate” in appendix A.5.1.

Switching on smart outlet based on pulse rate

A.5.1.2.1 Perform aerobic exercise or other activity which raises heart rate, while monitoring heart rate with an electronic heart rate monitor. Ensure heart rate increases past number entered in input field in step A.5.1.1.2 of procedure set “Configuring smart band for target heart rate” in appendix A.5.1. Proceed to step A.5.1.2.2.

A.5.1.2.2 Observe switching state of smart outlet. If smart outlet switches on, proceed to step A.5.1.3.1 of test procedure set “Switching off smart outlet based on pulse rate” in appendix A.5.1.

Switching off smart outlet based on pulse rate

A.5.1.3.1 Stop performing activity. Ensure heart rate decreases at least 10 units below number entered in input field in step A.5.1.1.2 of procedure set “Configuring smart band for target heart rate” in appendix A.5.1. Proceed to step A.5.1.3.2.

A.5.1.3.2 Observe switching state of smart outlet. If smart outlet switches off, test procedure is successful.

B Source code

All source listings have had copyright and license headers omitted. See appendix D for copyright and licensing information.

B.1 Network configuration

Listing 1 `init.lua`

```
22 enduser_setup.start(
23   function()
24     dofile("web.lua")
25   end,
26   function(err, str)
27     print("enduser_setup: Err #" .. err .. ": " .. str)
28   end,
29   print -- Lua print function can serve as the debug callback
30 );
```

B.2 Web interface

Listing 2 `web.lua`

```
27 local function connect (conn, data)
28    local query_data
29
30    conn:on ("receive",
31       function (cn, req_data)
32          query_data = get_http_req (req_data)
33          print (query_data["METHOD"] .. " " .. " " .. query_data["User-Agent"])
34          if(query_data["REQUEST"]["threshold"] == nil) then
35             srv = file.open("index.html", "r")
36             cn:send ("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n")
37             cn:send (srv:read())
38             srv:close()
39          elseif((tonumber(query_data["REQUEST"]["threshold"]) <= 48) or (tonumber(
                query_data["REQUEST"]["threshold"]) >= 180)) then
40             srv = file.open("error.html", "r")
41             cn:send ("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n")
42             cn:send (srv:read())
43             srv:close()
44          else
45             set = file.open("threshold", "w+")
46             set:write(query_data["REQUEST"]["threshold"])
47             cn:send ("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n")
48             cn:send ("<html><head><title>Smart band Web interface</title>")
49             cn:send ("</head><body><h1>Smart band Web interface</h1>")
50             cn:send ("<p>Setting has been submitted.</p>")
51             cn:send ("<p>Entered value: " .. query_data["REQUEST"]["threshold"] .. "</
                   p>")
52             cn:send ("<p>You may close this page now.</p>")
53             cn:send ("</body></html>")
54             set:close()
55             dofile("control.lua")
56          end
57       end)
58    conn:on ("sent",
```

```
59        function(cn)
60           cn:close ( )
61        end)
62  end
63
64  function wait_for_wifi_conn ( )
65     tmr.alarm (1, 1000, 1, function ( )
66        if wifi.sta.getip ( ) == nil then
67           print ("Waiting for Wifi connection")
68        else
69           tmr.stop (1)
70           print ("ESP8266 mode is: " .. wifi.getmode ( ))
71           print ("The module MAC address is: " .. wifi.ap.getmac ( ))
72           print ("Config done, IP is " .. wifi.sta.getip ( ))
73        end
74     end)
75  end
76
77  -- Build and return a table of the http request data
78  function get_http_req (instr)
79     local t = {}
80     local first = nil
81     local key, v, strt_ndx, end_ndx
82     local rk, rv
83
84     for str in string.gmatch (instr, "([^\n]+)") do
85        -- First line in the method and path
86        if (first == nil) then
87           first = 1
88           strt_ndx, end_ndx = string.find (str, "([^ ]+)")
89           v = trim (string.sub (str, end_ndx + 2))
90           key = trim (string.sub (str, strt_ndx, end_ndx))
91           t["METHOD"] = key
92           t["REQUEST"] = {}
93           if (v ~= nil) then
94              for rk, rv in string.gmatch (v, "(%w+)=(%w+)&*") do
95                  t["REQUEST"][rk] = rv
96              end
97           end
98        else -- Process and reamaining ":" fields
99           strt_ndx, end_ndx = string.find (str, "([^:]+)")
100          if (end_ndx ~= nil) then
101             v = trim (string.sub (str, end_ndx + 2))
102             key = trim (string.sub (str, strt_ndx, end_ndx))
103             t[key] = v
104          end
105       end
106    end
107
108    return t
109 end
110
111 -- String trim left and right
112 function trim (s)
113   return (s:gsub ("^%s*(.-)%s*$", "%1"))
114 end
```

```
115
116 -- Hang out until we get a wifi connection before the httpd server is started.
117 wait_for_wifi_conn ( )
118
119 -- Create the httpd server
120 svr = net.createServer (net.TCP, 30)
121
122 -- Server listening on port 80, call connect function if a request is received
123 svr:listen (8000, connect)
```

Listing 3 index.html

```
1  <html>
2  <head>
3  <title>Smart band Web interface</title>
4  </head>
5  <body>
6  <h1>Smart band Web interface</h1>
7  <p>This Web interface allows you to configure the heart rate you would like the smart
       band to switch upon crossing.</p>
8  <p>Enter a value between 48 and 180 BPM.</p>
9  <form action = "/" method="get">
10 Switching threshold:<br>
11 <input type="text" name="threshold">
12 <input type="submit" value="Set">
13 </form>
14 </body>
15 </html>
```

Listing 4 error.html

```
1  <html>
2  <head>
3  <title>Smart band Web interface</title>
4  </head>
5  <body>
6  <h1>Smart band Web interface</h1>
7  <p>The threshold you have entered is outside the valid range.
8  Please enter a threshold between 48 and 180 BPM.</p>
9  <p><a href="/">Return to threshold setting</a></p>
10 </form>
11 </body>
12 </html>
```

B.3 Pulse detection and smart outlet switching

Listing 5 control.lua

```
21 svr:close()
22
23 target = "192.168.1.100"
24
25 if file.open("threshold", "r") then
26     threshold = tonumber(file.read())
27     file.close()
28 end
29
30 pulse_count = 0;
```

```
31 first_timestamp = 0;
32 pulse_rate_bpm = 0;
33
34 gpio.mode(1,gpio.INT,gpio.PULLUP)
35 gpio.mode(12,gpio.INT,gpio.PULLUP)
36
37 udpSocket = net.createUDPSocket()
38 udpSocket:listen(5000)
39
40 gpio.trig(1,"down",function(pulse_level,current_timestamp)
41     if pulse_count == 0 then
42         pulse_count = 1
43         first_timestamp = current_timestamp
44     elseif pulse_count == 3 then
45         current_pulse_rate_bpm = (pulse_count*60*1000000*0.5) / (current_timestamp -
               first_timestamp);
46         if (current_pulse_rate_bpm >= 48) and (current_pulse_rate_bpm <= 180) then
47             if pulse_rate_bpm == 0 then
48                 pulse_rate_bpm = current_pulse_rate_bpm
49             else
50                 pulse_rate_bpm = (pulse_rate_bpm + current_pulse_rate_bpm) / 2
51             end
52         end
53         print("Pulse rate (BPM): " .. pulse_rate_bpm)
54         if pulse_rate_bpm > threshold then
55             print("Smart outlet on")
56             cmd = file.open("Smart_outlet_on.dat", "r")
57             udpSocket:send(80, target, cmd:read())
58             cmd.close()
59         else
60             print("Smart outlet off")
61             cmd = file.open("Smart_outlet_off.dat", "r")
62             udpSocket:send(80, target, cmd:read())
63             cmd.close()
64         end
65         pulse_count = 0
66     else
67         pulse_count = pulse_count + 1;
68     end
69     print("pulse: " .. pulse_count)
70 end)
71
72 gpio.trig(12,"down",function()
73     print("off")
74     pulse_count = 0
75     cmd = file.open("Smart_outlet_off.dat", "r")
76     udpSocket:send(80, target, cmd:read())
77     cmd.close()
78 end)
```

B.4 Smart outlet control commands

Hex dumps were generated from the binary files with `` `hexdump -C` `` on Linux.

Listing 6 `Smart_outlet_on.dat` (hex dump)

```
00000000  16 00 05 00 00 00 c9 1c  02 00 31 2e 36 2e 33 00  |..........1.6.3.|
00000010  45 43 4f 2d 37 38 30 35  37 45 34 45 00 00 00 00  |ECO-78057E4E....|
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000030  70 6c 75 67 00 00 00 00  00 00 00 00 00 00 00 00  |plug............|
00000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000050  37 38 30 35 37 45 34 45  00 00 00 00 00 00 00 00  |78057E4E........|
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000070  00 00 00 00 0b 7b fb 91  00 00 00 00 cd b8 42 2a  |.....{........B*|
00000080  01 01                                            |..|
00000082
```

Listing 7 `Smart_outlet_off.dat` (hex dump)

```
00000000  16 00 05 00 00 00 8d 16  02 00 31 2e 36 2e 33 00  |..........1.6.3.|
00000010  45 43 4f 2d 37 38 30 35  37 45 34 45 00 00 00 00  |ECO-78057E4E....|
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000030  70 6c 75 67 00 00 00 00  00 00 00 00 00 00 00 00  |plug............|
00000040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000050  37 38 30 35 37 45 34 45  00 00 00 00 00 00 00 00  |78057E4E........|
00000060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000070  00 00 00 00 bb 7a d9 99  00 00 00 00 77 f8 32 2c  |.....z......w.2,|
00000080  01 00                                            |..|
00000082
```

C Bill of materials

Vendor	Item #	Description	Quantity	Price
Amazon	B01DTUDPIG	HiLetgo Pulse sensor	1	$6.99
Amazon	B010O1G1ES	HiLetgo New Version NodeMCU	1	$8.79
Amazon	B01N3AHXKJ	Hatchbox 3D PETG filament	32.7 g	$0.92
Amazon	B00MRVJYEI	NETGEAR N300 Wi-Fi Router	1	$21.36
Tayda	A-204	LM311P IC Voltage Comparator	1	$0.23
Tayda	A-2100	7.5 kΩ 0.25 W 5% Carbon Film Resistor	1	$0.01
Tayda	A-604	10 kΩ 1-turn trimmer potentiometer	1	$0.19
Walmart	554470212	KAB Enterprise CT-065W WiFi Controlled Outlet	1	$19.84
			Total	**$58.33**

D Software licensing

init.lua

```
-- Copyright (c) 2016 Robert Foss
-- Modifications copyright (c) 2017 Siddharth Patil
--
-- Permission is hereby granted, free of charge, to any person obtaining a copy
-- of this software and associated documentation files (the "Software"), to deal
-- in the Software without restriction, including without limitation the rights
-- to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
-- copies of the Software, and to permit persons to whom the Software is
-- furnished to do so, subject to the following conditions:
--
-- The above copyright notice and this permission notice shall be included in all
-- copies or substantial portions of the Software.
--
-- THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
-- IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
-- FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
-- AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
-- LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
-- OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
-- SOFTWARE.
```

control.lua

```
-- Copyright (c) 2017 Siddharth Patil
--
-- Permission is hereby granted, free of charge, to any person obtaining a copy
-- of this software and associated documentation files (the "Software"), to deal
-- in the Software without restriction, including without limitation the rights
-- to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
-- copies of the Software, and to permit persons to whom the Software is
-- furnished to do so, subject to the following conditions:
--
-- The above copyright notice and this permission notice shall be included in all
-- copies or substantial portions of the Software.
--
-- THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
-- IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
-- FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
-- AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
-- LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
-- OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
-- SOFTWARE.
```

web.lua

```
-- Simple NodeMCU web server (done is a not so nodeie fashion :-)
-- (http://robotshop.com/letsmakerobots/nodemcu-esp8266-simple-httpd-web-server-
    example)
--
-- Written by Scott Beasley 2015
-- Open and free to change and use. Enjoy.
--
-- Modifications copyright (c) 2017 Siddharth Patil
```

```
--
-- Permission is hereby granted, free of charge, to any person obtaining a copy
-- of this software and associated documentation files (the "Software"), to deal
-- in the Software without restriction, including without limitation the rights
-- to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
-- copies of the Software, and to permit persons to whom the Software is
-- furnished to do so, subject to the following conditions:
--
-- The above copyright notice and this permission notice shall be included in all
-- copies or substantial portions of the Software.
--
-- THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
-- IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
-- FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
-- AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
-- LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
-- OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
-- SOFTWARE.
```

`index.html`, `error.html`, `Smart_outlet_on.dat`, `Smart_outlet_off.dat`

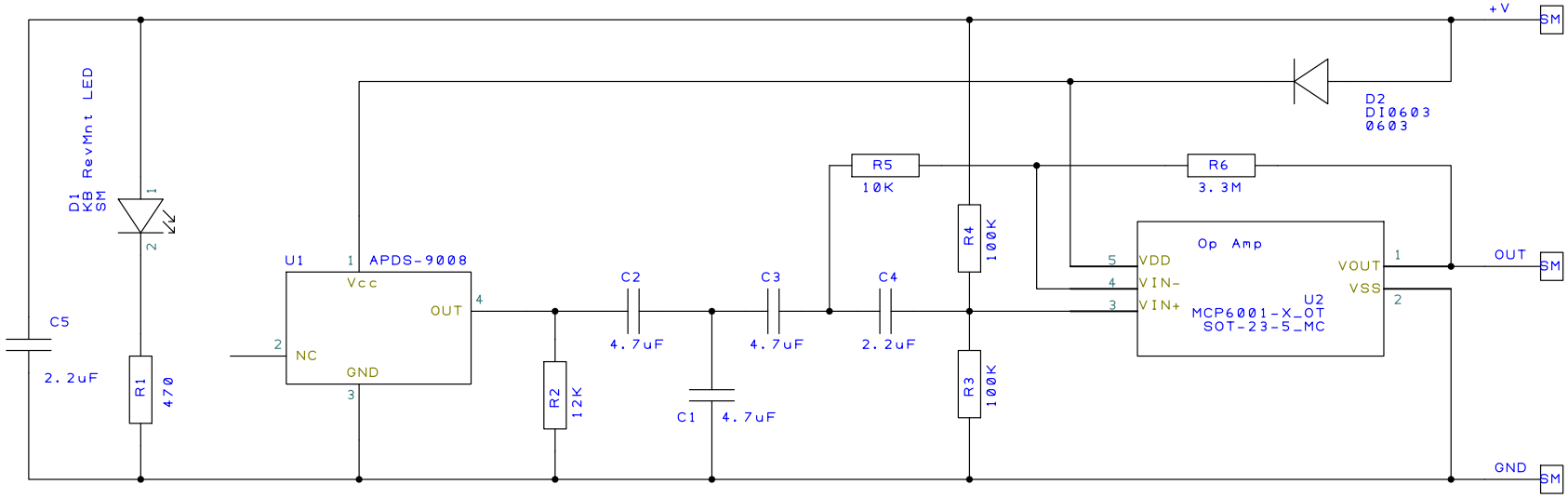
To the extent possible under law, the author(s) have dedicated all copyright and related and neighboring rights to this software to the public domain worldwide. This software is distributed without any warranty.

You should have received a copy of the CC0 Public Domain Dedication along with this software. If not, see <http://creativecommons.org/publicdomain/zero/1.0/>.

E Data sheets

Included datasheets:

- Pulse sensor schematic diagram
- TI LM311 datasheet, pages 1, 3, and 6
- ESP8266 datasheet, pages 6, 8



Pulse Sensor Amplified Designed by Joel Murphy Licensed under the TAPR Open Hardware License (www.tapr.org/OHL)

Spring 2012

LM111, LM211, LM311 Differential Comparators

1 Features

- Fast Response Time: 165 ns
- Strobe Capability
- Maximum Input Bias Current: 300 nA
- Maximum Input Offset Current: 70 nA
- Can Operate From Single 5-V Supply
- Available in Q-Temp Automotive
  - High-Reliability Automotive Applications
  - Configuration Control and Print Support
  - Qualification to Automotive Standards
- On Products Compliant to MIL-PRF-38535, All Parameters Are Tested Unless Otherwise Noted. On All Other Products, Production Processing Does Not Necessarily Include Testing of All Parameters.

2 Applications

- Desktop PCs
- Body Control Modules
- White Goods
- Building Automation
- Oscillators
- Peak Detectors

3 Description

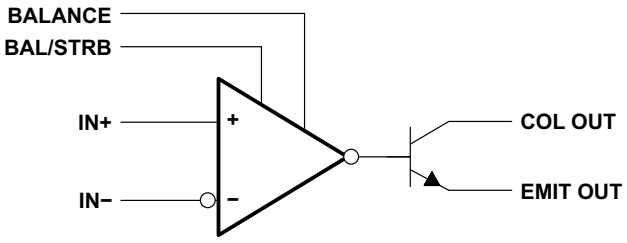
The LM111, LM211, and LM311 devices are single high-speed voltage comparators. These devices are designed to operate from a wide range of power-supply voltages, including ±15-V supplies for operational amplifiers and 5-V supplies for logic systems. The output levels are compatible with most TTL and MOS circuits. These comparators are capable of driving lamps or relays and switching voltages up to 50 V at 50 mA. All inputs and outputs can be isolated from system ground. The outputs can drive loads referenced to ground, V_{CC+} or V_{CC-}. Offset balancing and strobe capabilities are available, and the outputs can be wire-OR connected. If the strobe is low, the output is in the off state, regardless of the differential input.

Device Information(1)

PART NUMBER	PACKAGE	BODY SIZE
LM111FK	LCCC (20)	8.89 mm × 8.89 mm
LM111JG	CDIP (8)	9.60 mm × 6.67 mm
LM311PS	SO (8)	6.20 mm × 5.30 mm
LM211D	SOIC (8)	4.90 mm × 3.91 mm
LM311D		
LM211P	PDIP (8)	9.81 mm × 6.35 mm
LM311P		
LM211PW	TSSOP (8)	3.00 mm × 4.40 mm
LM311PW		

(1) For all available packages, see the orderable addendum at the end of the data sheet.

Simplified Schematic

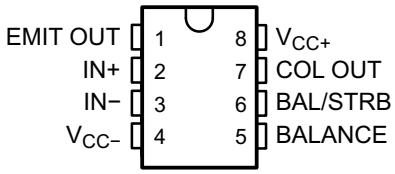


Copyright © 2016, Texas Instruments Incorporated

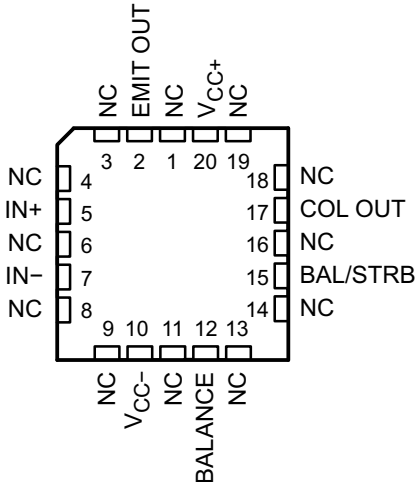
An IMPORTANT NOTICE at the end of this data sheet addresses availability, warranty, changes, use in safety-critical applications, intellectual property matters and other important disclaimers. PRODUCTION DATA.

5 Pin Configuration and Functions

LMx11 D, JG, P, PS, or PW Package
8-Pin SOIC, CDIP, PDIP, SO or TSSOP
Top View



LM111 FK Package
20-Pin LCCC(1)
Top View



(1) NC = No internal connection

Pin Functions

PIN					I/O(1)	DESCRIPTION
NAME	LM211, LM311	LM311	LM111	LM111		
	SOIC, PDIP, TSSOP	SO	CDIP	LCCC		
IN+	2	2	2	5	I	Noninverting comparator
IN–	3	3	3	7	I	Inverting input comparator
BALANCE	5	5	5	12	I	Balance
BAL/STRB	6	6	6	15	I	Strobe
COL OUT	7	7	7	17	O	Output collector comparator
EMIT OUT	1	1	1	2	O	Output emitter comparator
V_{CC}–	4	4	4	10	—	Negative supply
V_{CC}+	8	8	8	20	—	Positive supply
NC	—	—	—	1	—	No connect (No internal connection)
				3		
				4		
				6		
				8		
				9		
				11		
				13		
				14		
				16		
				18		
				19		

(1) I = Input, O = Output

Copyright © 1973–2017, Texas Instruments Incorporated

6.6 Electrical Characteristics

at specified free-air temperature, $V_{CC\pm}$ = ±15 V (unless otherwise noted)

PARAMETER		TEST CONDITIONS		T_A (1)	LM111, LM211, LM211Q			LM311			UNIT
					MIN	TYP (2)	MAX	MIN	TYP (2)	MAX	
V_{IO}	Input offset voltage	See (3)		25°C		0.7	3		2	7.5	mV
				Full range			4			10	
I_{IO}	Input offset current	See (3)		25°C		4	10		6	50	nA
				Full range			20			70	
I_{IB}	Input bias current	1 V ≤ V_O ≤ 14 V		25°C		75	100		100	250	nA
				Full range			150			300	
$I_{IL(S)}$	Low-level strobe current (4)	$V_{(strobe)}$ = 0.3 V, V_{ID} ≤ –10 mV		25°C		–3			–3		mA
V_{ICR}	Common-mode input-voltage range (3)	Lower range		Full range		–14.7	–14.5		–14.7	–14.5	V
		Upper range			13	13.8		13	13.8		
A_{VD}	Large-signal differential-voltage amplification	5 V ≤ V_O ≤ 35 V, R_L = 1 kΩ		25°C	40	200		40	200		V/mV
I_{OH}	High-level (collector) output leakage current	$I_{(strobe)}$ = –3 mA, V_{ID} = 5 mV	V_{OH} = 35 V	25°C		0.2	10				nA
				Full range			0.5				µA
		V_{ID} = 5 mV, V_{OH} = 35 V		25°C					0.2	50	nA
V_{OL}	Low-level (collector-to-emitter) output voltage	I_{OL} = 50 mA	V_{ID} = –5 mV	25°C		0.75	1.5				V
			V_{ID} = –10 mV	25°C					0.75	1.5	
		V_{CC+} = 4.5 V, V_{CC-} = 0 V, I_{OL} = 8 mA	V_{ID} = –6 mV	Full range		0.23	0.4				
			V_{ID} = –10 mV	Full range					0.23	0.4	
$I_{CC}+$	Supply current from V_{CC+} output low	V_{ID} = –10 mV,	No load	25°C		5.1	6		5.1	7.5	mA
$I_{CC}–$	Supply current from V_{CC-} output high	V_{ID} = 10 mV,	No load	25°C		–4.1	–5		–4.1	–5	mA

(1) Unless otherwise noted, all characteristics are measured with BALANCE and BAL/STRB open and EMIT OUT grounded. Full range for LM111 is –55°C to 125°C, for LM211 is –40°C to 85°C, for LM211Q is –40°C to 125°C, and for LM311 is 0°C to 70°C.
(2) All typical values are at T_A = 25°C.
(3) The offset voltages and offset currents given are the maximum values required to drive the collector output up to 14 V or down to 1 V with a pullup resistor of 7.5 kΩ to V_{CC+}. These parameters actually define an error band and take into account the worst-case effects of voltage gain and input impedance.
(4) The strobe must not be shorted to ground; it must be current driven at –3 mA to –5 mA (see Figure 18 and Figure 31).

6.7 Switching Characteristics

$V_{CC\pm}$ = ±15 V, T_A = 25°C

PARAMETER	TEST CONDITIONS	LM111, LM211, LM211Q, LM311	UNIT
		TYP	
Response time, low-to-high-level outputSee (1)	R_C = 500 Ω to 5 V, C_L = 5 pF, see (2)	115	ns
Response time, high-to-low-level outputSee (1)		165	ns

(1) The response time specified is for a 100-mV input step with 5-mV overdrive and is the interval between the input step function and the instant when the output crosses 1.4 V.
(2) The package thermal impedance is calculated in accordance with MIL-STD-883.

Copyright © 1973–2017, Texas Instruments Incorporated

1. General Overview

1.1. Introduction

Espressif Systems' Smart Connectivity Platform (ESCP) is a set of high performance, high integration wireless SOCs, designed for space and power constrained mobile platform designers. It provides unsurpassed ability to embed WiFi capabilities within other systems, or to function as a standalone application, with the lowest cost, and minimal space requirement.

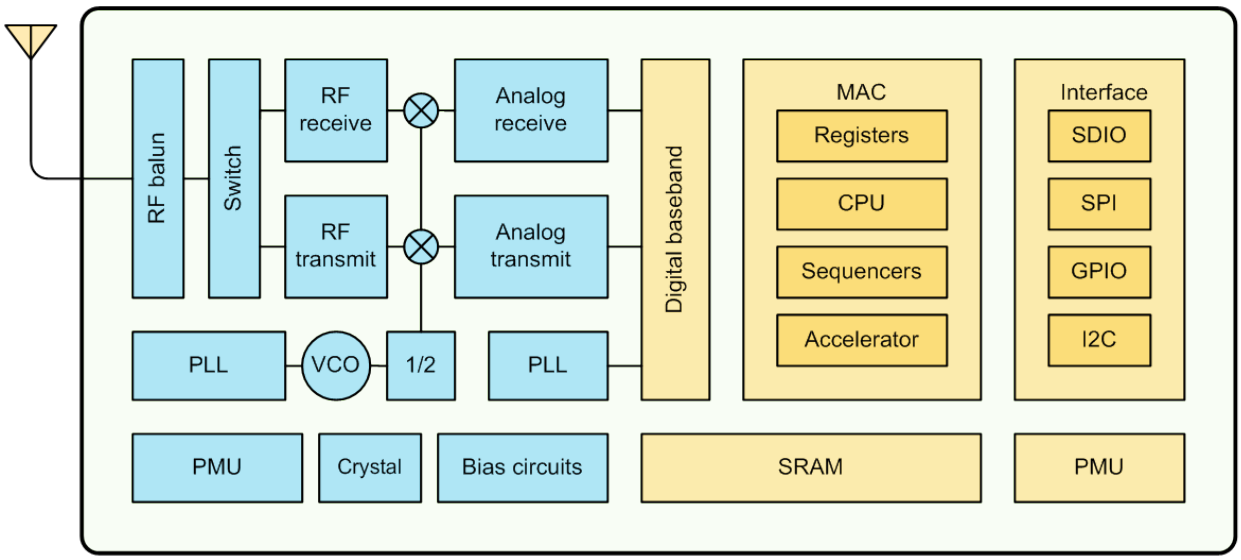


Figure 1 ESP8266EX Block Diagram

ESP8266EX offers a complete and self-contained WiFi networking solution; it can be used to host the application or to offload WiFi networking functions from another application processor.

When ESP8266EX hosts the application, it boots up directly from an external flash. In has integrated cache to improve the performance of the system in such applications.

Alternately, serving as a WiFi adapter, wireless internet access can be added to any micro controller-based design with simple connectivity (SPI/SDIO or I2C/UART interface).

ESP8266EX is among the most integrated WiFi chip in the industry; it integrates the antenna switches, RF balun, power amplifier, low noise receive amplifier, filters, power management modules, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.

ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor, with on-chip SRAM, besides the WiFi functionalities. ESP8266EX is often integrated with external sensors and other application specific devices through its GPIOs; sample codes for such applications are provided in the software development kit (SDK).

Categories	Items	Values
WiFi Paramters	Certificates	FCC/CE/TELEC/SRRC
	WiFi Protocles	802.11 b/g/n
	Frequency Range	2.4G-2.5G (2400M-2483.5M)
	Tx Power	802.11 b: +20 dBm
		802.11 g: +17 dBm
		802.11 n: +14 dBm
	Rx Sensitivity	802.11 b: -91 dbm (11 Mbps)
		802.11 g: -75 dbm (54 Mbps)
		802.11 n: -72 dbm (MCS7)
	Types of Antenna	PCB Trace, External, IPEX Connector, Ceramic Chip
Hardware Paramaters	Peripheral Bus	UART/SDIO/SPI/I2C/I2S/IR Remote Control
		GPIO/PWM
	Operating Voltage	3.0~3.6V
	Operating Current	Average value: 80mA
	Operating Temperature Range	-40°~125°
	Ambient Temperature Range	Normal temperature
	Package Size	5x5mm
	External Interface	N/A
Software Parameters	WiFi mode	station/softAP/SoftAP+station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network)
	Ssoftware Development	Supports Cloud Server Development / SDK for custom firmware development
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP