Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

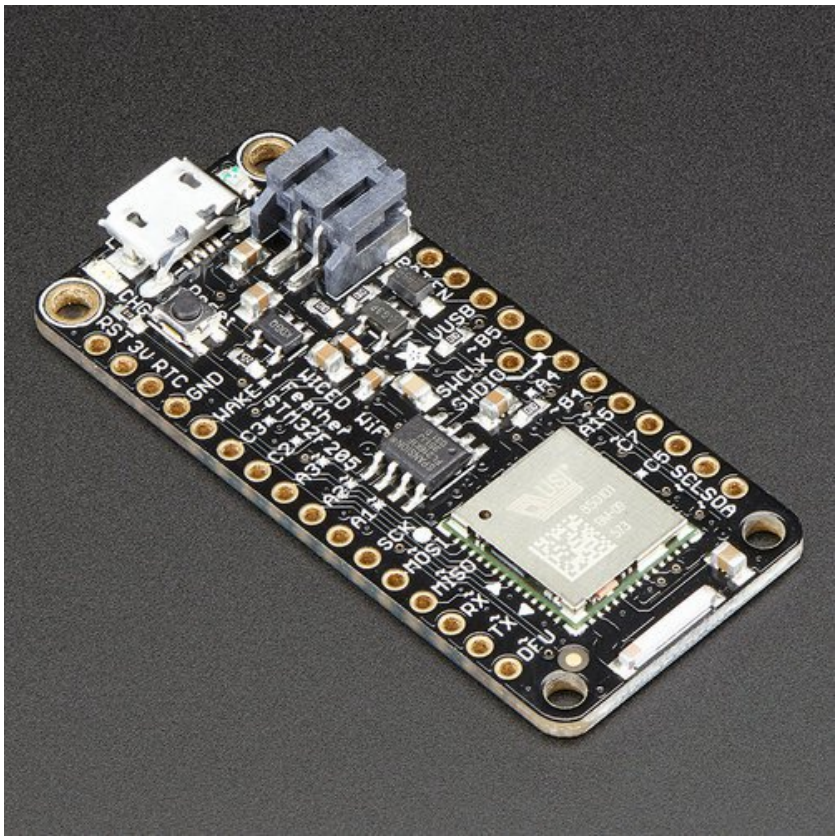
We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Introducing the Adafruit WICED Feather WiFi

Created by Kevin Townsend



Last updated on 2017-04-27 07:20:12 PM UTC

Guide Contents

Guide Contents 2
Overview 14
Board Layout 17
Pin Multiplexing 17
- Accessing Pins in Software 17

Power Config 18
- LIPO Cell Power Monitoring (A1) 19

16 Mbit (2MByte) SPI Flash 19
PWM Outputs 20
Assembly 21
Header Options! 21
Soldering in Plain Headers 23
- Prepare the header strip: 23
- Add the breakout board: 23
- And Solder! 24

Soldering on Female Header 25
- Tape In Place 25
- Flip & Tack Solder 26
- And Solder! 26

Get the WICED BSP 28
Adding Adafruit Board Support 28
- Add the Adafruit BSP List 28
- Add the Adafruit WICED BSP 29

Upgrading From Earlier WICED BSP Releases (<0.6.0) 29
Windows Setup 30
Install Adafruit Windows Drivers 30
Install libusb 0.1 Runtime 30
Install Python 2.7 31
- Testing the Python Installation 31

Install Python Tools 32
- Testing the Installation 32

Optional: Install AdaLink 32
OS X Setup 33

Install dfu-util 33
Testing the Installation 33
Install Python Tools 33
Testing the Installation 33
Optional: Install AdaLink 34
Linux Setup 35
UDEV Setup 35
Install dfu-util 35
Building dfu-util From Source (Ubuntu 14.04 etc.) 35
Testing the Installation 36
Install Python Tools 36
Testing the Installation 36
Optional: Install AdaLink 36
External Resources 36
Arduino IDE Setup 38
Board Selection 38
Setting the 'Section' 38
Selecting the Serial Port 39
Optional: Updating the Bootloader 40
Compiling your Sketch 40
System Architecture 42
WICED WiFi + RTOS + SDEP = FeatherLib 42
Arduino User Code 42
Inter Process Communication (SDEP) 42
Flash Memory Layout 42
User Code (256KB + 20KB SRAM) 43
Feather Lib (704 KB + 108KB SRAM) 43
Config Data (32KB) 43
USB DFU Bootloader (32KB) 43
USB Setup 43
DFU Mode (Fast Blinky) 43
Normal Operating Mode (User Code) 43
Flash Updates 44
WICED Feather API 45

AdafruitFeather 45
AdafruitTCP 45
AdafruitUDP 45
AdafruitHTTP 45
AdafruitMQTT 45
AdafruitAIO 45
AdafruitSDEP 45
Client API 46
AdafruitFeather 47
AdafruitFeather API 47
Firmware Version Management 47
char const* bootloaderVersion (void) 47
char const* sdkVersion (void) 48
char const* firmwareVersion (void) 48
char const* arduinoVersion (void) 48
Scanning 48
int scanNetworks (wl_ap_info_t ap_list[], uint8_t max_ap) 48
Connecting 48
bool connect (void) 48
bool connect (const char *ssid) 48
bool connect (const char *ssid, const char *key, int enc_type = ENC_TYPE_AUTO) 49
bool begin (void) 49
bool begin (const char *ssid) 49
bool begin (const char *ssid, const char *key, int enc_type = ENC_TYPE_AUTO) 49
void disconnect (void) 49
Network and Connection Details 49
bool connected (void); 49
uint8_t* macAddress (uint8_t *mac); 49
uint32_t localIP (void); 50
uint32_t subnetMask (void); 50
uint32_t gatewayIP (void); 50
char* SSID (void); 50
int32_t RSSI (void); 50
int32_t encryptionType (void); 50
uint8_t* BSSID (uint8_t* bssid); 50
DNS Lookup 50

IPAddress hostByName (const char* hostname) 50
bool hostByName (const char* hostname, IPAddress& result) 51
bool hostByName (const String &hostname, IPAddress& result) 51
Ping 51
uint32_t ping (char const* host) 51
uint32_t ping (IPAddress ip) 51
Factory Reset 51
void factoryReset (void) 51
void nvmReset (void) 52
Hardware Random Number Generator 52
bool randomNumber (uint32_t* random32bit) 52
Real Time Clock 52
bool getISO8601Time (iso8601_time_t* iso8601_time) 52
uint32_t getUtcTime (void) 53
TLS Root Certificate Management 53
Default Root Certificates 53
bool useDefaultRootCA (bool enabled) 53
bool initRootCA (void) 54
bool addRootCA (uint8_t const* root_ca, uint16_t len) 54
bool clearRootCA (void) 54
Print Helpers 54
void printVersions (Print& p = Serial) 54
void printNetwork (Print& p = Serial) 54
void printEncryption (int32_t enc, Print& p = Serial) 55
AdafruitFeather: Profiles 56
Connecting via Profiles 56
Profiles API 56
bool saveConnectedProfile (void) 56
bool addProfile (char* ssid) 56
bool addProfile (char* ssid, char* key, wl_enc_type_t enc_type) 56
bool removeProfile (char* ssid) 57
void clearProfiles (void) 57
char* profileSSID (uint8_t pos); 57
int32_t profileEncryptionType (uint8_t pos); 57
AdafruitTCP 59
TCP Socket API 59
Packet Buffering 59

void usePacketBuffering (bool enable) 59
TLS/SSL Certificate Verification 59
Verifying Certificates with the WICED Feather (Safer) 60
Ignoring Certificate Verification (Easier) 60
Default Root Certificates 60
void tlsRequireVerification (bool required) 60
Socket Handler Functions 61
void getHandle (void) 61
Client API 61
int connect (IPAddress ip, uint16_t port) 61
int connect (const char * host, uint16_t port) 61
int connectSSL (IPAddress ip, uint16_t port) 61
int connectSSL (const char* host, uint16_t port) 61
uint8_t connected (void) 62
void stop (void) 62
Stream API 62
int read (void) 62
int read (uint8_t * buf, size_t size) 62
size_t write (uint8_t data) 62
size_t write (const uint8_t *content, size_t len) 62
int available (void) 63
int peek (void) 63
void flush (void) 63
Callback API 63
void setReceivedCallback (tcpcallback_t fp) 63
void setDisconnectCallback (tcpcallback_t fp) 63
Callback Function Signatures 63
Example: Callback Based HTTP Request 64
AdafruitTCPServer 66
Constructor 66
Functions 66
bool begin (void) 66
AdafruitTCP accept (void) 66
AdafruitTCP available (void) 66
void stop (void) 66
void setConnectCallback (tcpserver_callback_t fp) 66
Example 67
AdafruitUDP 69

UDP Socket API 69
UDP API 69
uint8_t begin (uint16_t port) 69
void stop (void) 69
int beginPacket (IPAddress ip, uint16_t port) 69
int beginPacket (const char *host, uint16_t port) 70
int endPacket (void) 70
int parsePacket (void) 70
IPAddress remoteIP (void) 70
uint16_t remotePort (void) 70
Stream API 70
int read (void) 70
int read (unsigned char* buffer, size_t len) int read (char* buffer, size_t len) 70
int peek (void) 71
int available (void) 71
void flush (void) 71
size_t write (uint8_t byte) 71
size_t write (const uint8_t *buffer, size_t size) 71
Callback Handlers 71
void setReceivedCallback (udpcallback_t fp) 72
Examples 72
UDP Echo Server 72
AdafruitHTTP 74
AdafruitHTTP API 74
HTTP Headers 74
bool addHeader (const char* name, const char* value) 74
bool clearHeaders (void) 74
HTTP GET Requests 74
bool get (char const* url) 74
bool get (char const* host, char const* url) 74
HTTP POST Requests 75
bool post (char const* url, char const* encoded_data) 75
bool post (char const* host, char const* url, char const* encoded_data) 75
HTTP GET Example 75
AdafruitHTTPServer 78
AdafruitHTTPServer API 78

Constructor 78
Adding Pages 78
1. HTTPPageRedirect Records (Page Redirection Entries) 78
2. HTTPPage Records (Standard Pages) 79
Converting Static Content (HTTPResources) 79
Implementing Dynamic Page Handlers 79
Registering the Pages 80
Starting/Stopping the HTTP Server 80
Complete Example 80
AdafruitMQTT 84
Constructors 84
Functions 84
Connection Management 84
bool connected(void) 84
bool connect (IPAddress ip, uint16_t port = 1883, bool cleanSession = true, uint16_t keepalive_sec = MQTT_KEEPALIVE_DEFAULT); 85
bool connect (const char* host, uint16_t port = 1883, bool cleanSession = true, uint16_t keepalive_sec = MQTT_KEEPALIVE_DEFAULT); 85
bool connectSSL (IPAddress ip, uint16_t port = 8883, bool cleanSession = true, uint16_t keepalive_sec = MQTT_KEEPALIVE_DEFAULT) 85
bool connectSSL (const char* host, uint16_t port = 8883, bool cleanSession = true, uint16_t keepalive_sec = MQTT_KEEPALIVE_DEFAULT) 86
bool disconnect (void) 86
Messaging 86
bool publish (UTF8String topic, UTF8String message, uint8_t qos = MQTT_QOS_AT_MOST_ONCE, bool retained = false); 86
bool subscribe (const char* topicFilter, uint8_t qos, messageHandler mh); 87
Subscribe Callback Handler(s) 87
Callback Handler Parameters 87
bool unsubscribe(const char* topicFilter); 88
Last Will 88
void will (const char* topic, UTF8String message, uint8_t qos = MQTT_QOS_AT_MOST_ONCE, uint8_t retained = 0); 88
Client ID 88
void clientID(const char* client) 88
Disconnect Callback 88
AdafruitMQTT Example 89
AdafruitMQTTTopic 93
Constructor 93

Functions 93
void retain (bool on) 93
Subscribe Callbacks 93
bool subscribe (messageHandler_t mh) 94
bool unsubscribe (void) 94
bool subscribed (void) 94
Publishing Data via 'Print' 94
Example 95
AdafruitAIO 98
Constructor 98
Functions 98
Connecting 98
bool connect (bool cleanSession = true, uint16_t keepalive_sec = MQTT_KEEPALIVE_DEFAULT) 98
bool connectSSL (bool cleanSession = true, uint16_t keepalive_sec = MQTT_KEEPALIVE_DEFAULT) 99
Feed Management 99
bool updateFeed (const char* feed, UTF8String message, uint8_t qos=MQTT_QOS_AT_MOST_ONCE, bool retain=true) 99
bool followFeed (const char* feed, uint8_t qos, messageHandler_t mh) 99
bool unfollowFeed (const char* feed) 100
Example 100
AdafruitAIOFeed 104
Constructor 104
Functions 104
bool follow (feedHandler_t fp) 104
bool unfollow (void) 104
bool followed (void) 104
Example 105
AdafruitTwitter 108
1. Creating a WICED Twitter Application 108
Enter the Application Details 108
Set the Application Permissions 109
Manage the Access Keys 109
Copy the Appropriate Key Data 109
Create your Access Token 110

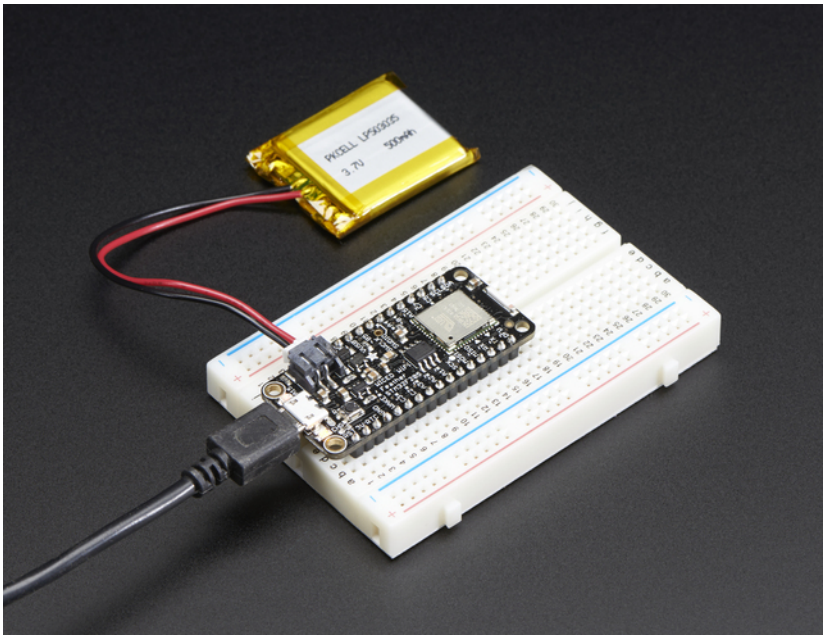
2. Using the AdafruitTwitter Class 110
AdafruitSDEP 113
AdafruitSDEP API 113
Constructor 113
Functions 113
sdep 113
Examples 114
sdep_n 114
Examples 114
Error Handling Functions 115
err_t errno (void) 115
char const* errstr(void) 115
char const* cmdstr (uint16_t cmd_id) 115
void err_actions (bool print, bool halt) 116
Error Handling Example 116
Client 117
Adapting Client Examples 117
1. Update Header Includes 117
2. Change 'WiFi.*' References to 'Feather.*' 117
3. Change WiFiUDP and WiFiTCP Class Types 118
Constants 119
wl_enc_type_t 119
err_t 119
wl_ap_info_t 120
Python Tools 121
pyresource.py (Convert static files to C headers) 121
pycert.py (Python TLS Certificate Converter) 121
feather_dfu.py (Python USB DFU Utility) 121
pyresource.py 122
Usage 122
HTTPResource Records 122
HTTPResource Collection: resources.h 123
pycert.py 124
Downloading the Root Certificate for a Domain 124
Parameters 124

Usage 124
Converting PEM Files 124
Parameters 124
Usage 125
feather_dfu.py 126
Commands 126
arduino_upgrade 126
featherlib_upgrade 126
enter_dfu 126
info 126
factory_reset 127
nvm_reset 127
reboot 127
SDEP Commands 128
Generic 130
Reset (0x0001) 130
Factory Reset (0x0002) 130
Enter DFU Mode (0x0003) 130
System Information (0x0004) 130
Parameter ID 130
NVM Reset (0x0005) 131
Error String (0x0006) 131
Error ID 131
Generate Random Number (0x0101) 131
Examples 133
Accessing the Examples (Arduino 1.6.5) 133
Accessing the Examples (Arduino >= 1.6.8) 133
Example Folders 133
Making Modifications to the Examples 133
ScanNetworks 134
Setup 134
Compile and Flash 134
Testing the Sketch 134
Ping 135

Setup 135
Compile and Flash 135
Testing the Sketch 135
GetHostByName 137
Setup 137
Compile and Flash 137
Testing the Sketch 137
HttpGetPolling 139
Setup 139
Compile and Flash 139
Testing the Sketch 139
HttpGetCallback 141
Setup 141
Compile and Flash 141
Testing the Sketch 141
HTTPSLargeData 143
Setup 143
Compile and Flash 143
Testing the Sketch 143
Throughput 145
Setup 145
Running Netcat 145
Compile and Flash 145
Testing the Sketch 145
FeatherOLED 147
Setup 147
Setting the Access Point 147
Enabling LIPO Battery Monitoring (Optional) 147
Enabling the TSL2561 Luminosity Sensor (Optional) 147
Enabling MQTT to Adafruit IO (Optional) 148
Compile and Flash 148
Testing the Sketch 148
FAQs 149

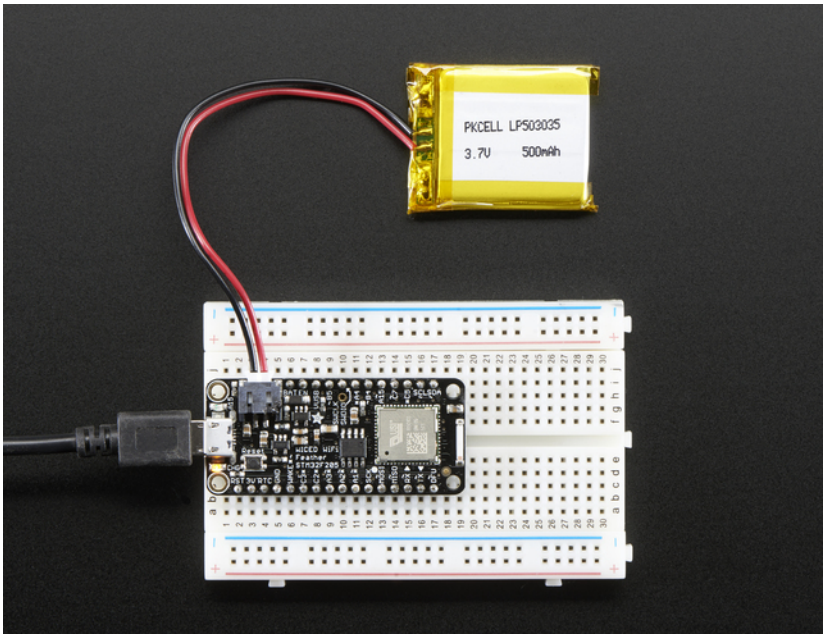
Downloads 152
Related Documents 152
Schematic 152
Fabrication Print 152

Overview



Feather (http://adafru.it/l7B) is the new development board from Adafruit, and like its namesake it is thin, light, and lets you fly! We designed Feather to be a new standard for portable microcontroller cores. This is the **Adafruit WICED Feather** - it's our most powerful Feather yet!We have other boards in the Feather family, check'em out here. (http://adafru.it/l7B)

Say "Hi!" the WICED Feather! Perfect for your next Internet connected project, with a processor and WiFi core that can take anything you throw at it!



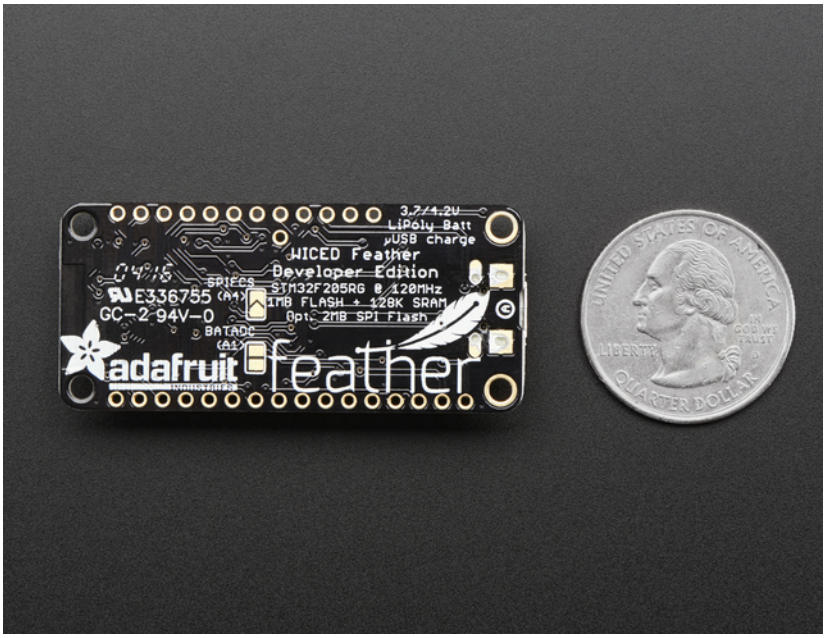
The WICED Feather is based on Broadcom's WICED (Wireless Internet Connectivity for Embedded Devices) platform, and is paired up with a powerful STM32F205 ARM Cortex M3 processor running at 120MHz, with support for TLS 1.2 to access sites and web services safely and securely.

We spent a lot of time adding support for this processor and WiFi chipset to the Arduino IDE you know and love. Programming doesn't rely on any online or third party tools to build, flash or run your code. You write your code in the Arduino IDE using many of the same standard libraries you've always used (Wire, SPI, etc.), compile locally, and the device is flashed directly from the IDE over USB. **Note that this chipset is not**

identical to the Arduino standard-supported Atmega series and many libraries that are written for AVR will not compile or work with WICED!

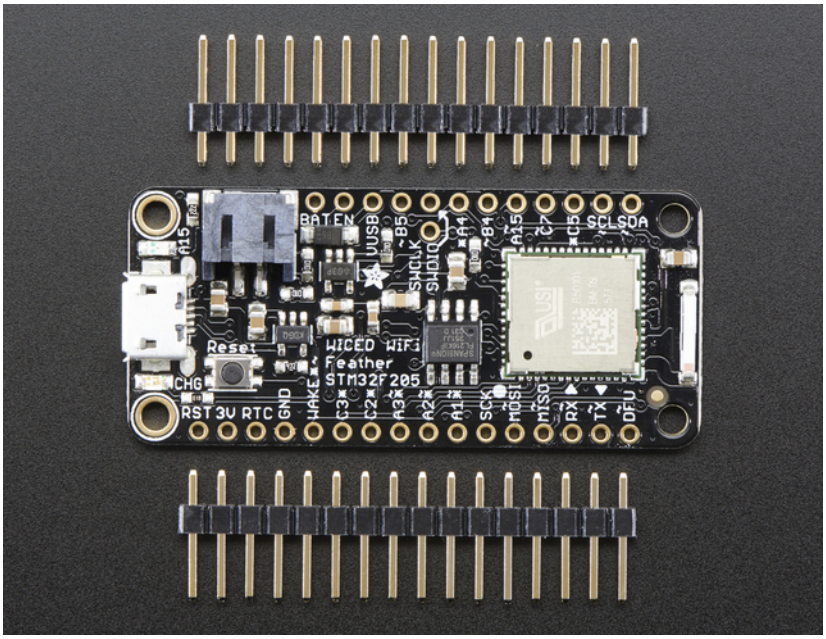
Since the WICED Feather is based on the standard Adafruit Feather (http://adafru.it/mf2) layout, you also have instant access to a variety of Feather Wings, as well as all the usual standard breakouts available from Adafruit or other vendors.

After more than a year of full time effort in the making, we think it's the best and most flexible WiFi development board out there, and the easiest way to get your TCP/IP-based project off the ground without sacrificing flexibility or security. We even cooked in some built-in libraries in the WiFi core, wuch as TCP client and Server, HTTP client and server, and MQTT client (with easy Adafruit IO interfacing).



The WICED Feather has the following key features:

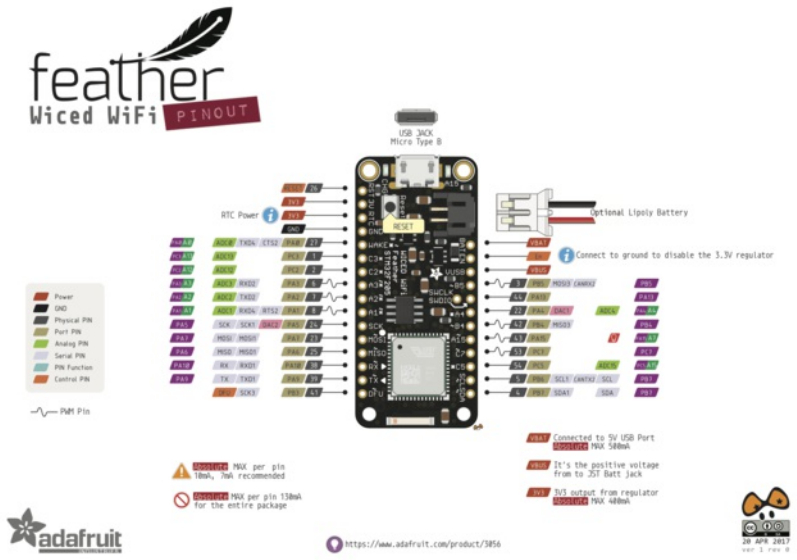
- Measures 2.0" x 0.9" x 0.28" (51mm x 23mm x 8mm) without headers soldered in
- Light as a (large?) feather - 5.7 grams
- STM32F205RG (http://adafru.it/m9A) 120MHz ARM Cortex M3 MCU
- BCM43362 (http://adafru.it/meC) 802.11b/G/N radio
- 128KB SRAM and 1024KB flash memory (total)
- 16KB SRAM and 128KB flash available for user code
- 16MBit (2MB) SPI flash for additional data storage
- Built in Real Time Clock (RTC) with optional external battery supply
- Hardware SPI and I2C (including clock-stretching)
- 12 standard GPIO pins, with additional GPIOs available via SPI, UART and I2C pins
- 7 standard PWM outputs, with additional outputs available via SPI, UART and I2C pins
- Up to 8 12-bit ADC inputs
- Two 12-bit DAC outputs (Pins A4 and SCK/A5)
- Up to 3 UARTs (including one with full HW flow control)
- TLS 1.2 support to access secure HTTPS and TCP servers
- On board single-cell LIPO charging and battery monitoring
- Fast and easy firmware updates to keep your module up to date
- Based on the excellent community-supported Maple (http://adafru.it/mpE) project



Comes fully assembled and tested, with a USB bootloader that lets you quickly use it with the Arduino IDE. We also toss in some headers so you can solder it in and plug into a solderless breadboard. **Lipoly battery (http://adafru.it/e0v) and MicroUSB cable (http://adafru.it/aM5) not included** (but we do have lots of options in the shop if you'd like!)

Our learn guide will show you everything you need to know to get your projects online, and connected to the outside world!

Board Layout

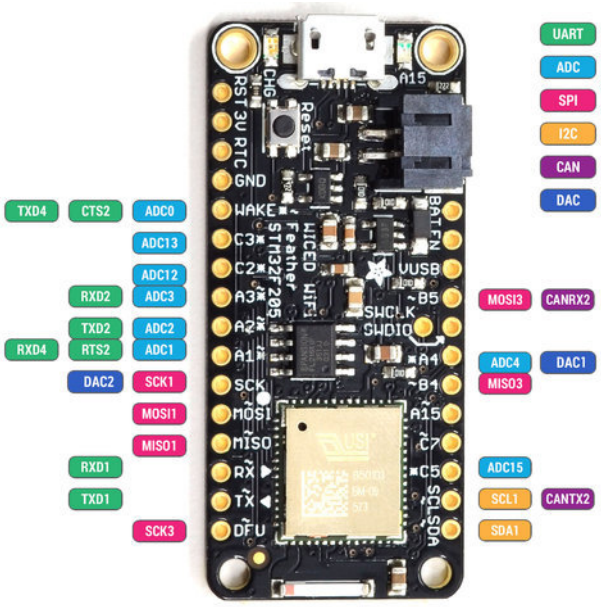


The WICED Feather uses the same standard pinout as the rest of the[Feather family](http://adafru.it/m0b) (http://adafru.it/m0b), allowing you to use the same Feather Wings across all your compatible devices.

It has the standard Feather on board LIPO battery charger (simply connect a LIPO battery and USB power at the same time), and 3.3V voltage regulation from either USB or VBAT (the LIPO cell) with automatic switching between power supplies.

Pin Multiplexing

The pins on the WICED Feather can be configured for several different purposes, with the main config options shown in the illustration below:



Accessing Pins in Software

For most pin names, you **must append 'P'** to the pin name shown on the silk screen. The table below lists the pin names on the silkscreen and their corresponding macro in your Arduino code:

Slikscreen	Arduino Code	Note(s)
WAKE	WAKE or PA0	-
C3	PC3	-
C2	PC2	-
A3	PA3	-
A2	PA2	-
A1	PA1	-
SCK	SCK or PA5	-
MOSI	MOSI or PA7	-
MISO	MISO or PA6	-
RX	PA10	-
TX	PA9	-
DFU	PB3	-
B5	PB5	-
SWCLK	PA14	-
SWDIO	PA13	-
A4	P14	-
B4	PB4	-
A15	PA15	-
C7	PC7	-
C5	PC5	-
SCL	PB6	-
SDA	PB7	-

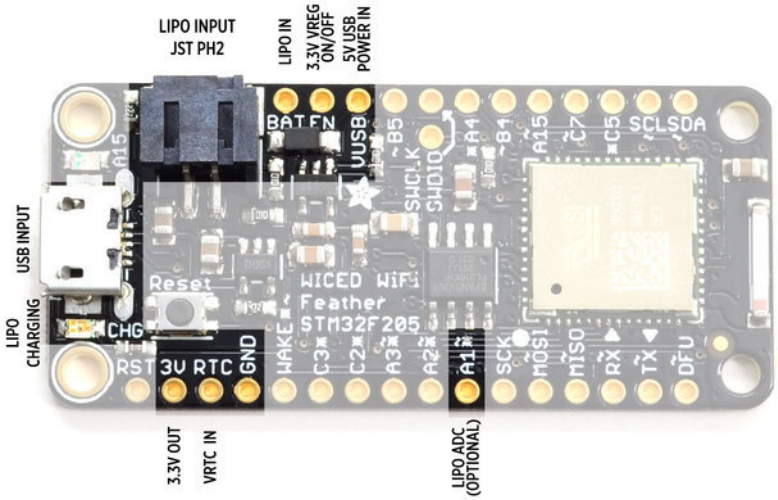
Other notable pins defined in[feather.h](http://adafru.it/olD) include:

Main Macro Name	Direct Arduino Pin Name
BOARD_LED_PIN	PA15

For further details on the board layout, see the[schematic here](http://adafru.it/olE).

Power Config

The WICED Feather can be run from either 5V USB power or a standard ~3.7V LIPO cell, and includes the ability to charge LIPO cells from USB power when both are connected at the same time.



The following pins are included as part of the WICED Feather's power system:

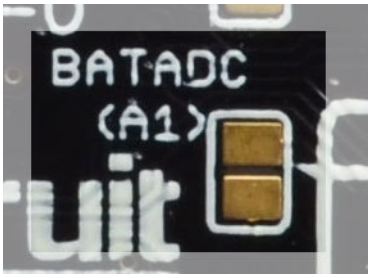
- **3V**: The output of the on-board 3.3V 600mA voltage regulator
- **RTC**: The input for the realt-time clock (RTC) on the STM32F205 (optional)
- **GND**: The common/GND pin which should be connect to GND on any other boards you use

- **BAT**: The input for the 3.7V LIPO cell
- **EN**: The 'EN' switch for the 3.3V voltage regulator. Set this to GND to disable power.
- **VUSB**: The 5V USB power input (USB VBUS)
- **A1**: This pin is optionally connected to a 10K+10K voltage divider that allows you to safely measure the output of the LIPO cell using the internal ADC (analog to digital converter).

LIPO Cell Power Monitoring (A1)

The LIPO battery level can optionally be monitored via a voltage divider configured on ADC pin **A1**.

To enable the 10K + 10K voltage divider (which will divide the LIPO voltage levels in half so that the ADC pin can safely read them), you need to solder shut the **BATADC** solder jumper on the bottom of the PCB:



This will allow you to read the voltage level of the LIPO cell using pin **A1** where each value on the ADC is equal to **0.80566mV** since:

- 3300mV / 4096 (12-bit ADC) = 0.80566406mV per LSB

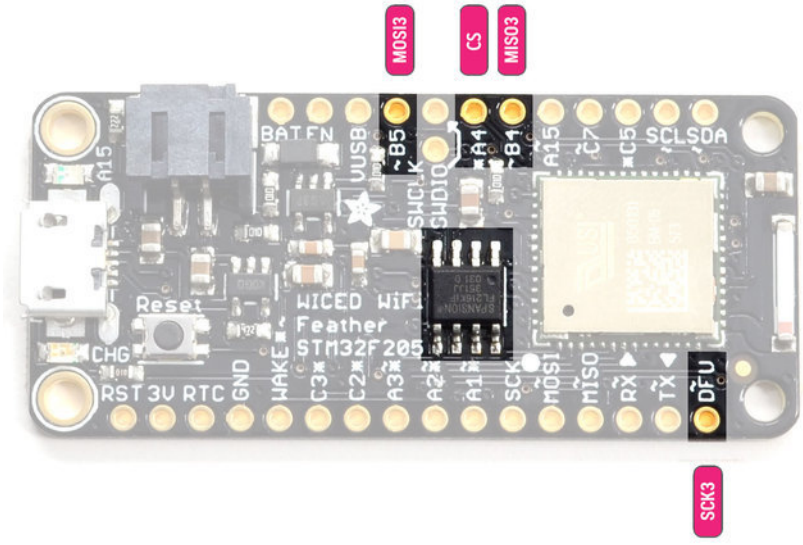
You need to **double** the calculated voltage to compensate for the 10K+10K voltage divider, so in reality every value from the ADC is equal to **1.61133mV** on the LIPO cell, although it appears on the ADC at half that level.

16 Mbit (2MByte) SPI Flash

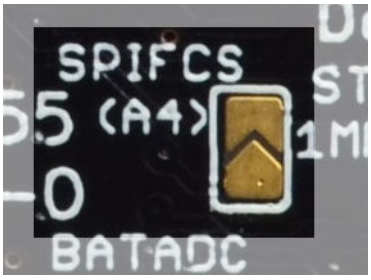
The WICED Feather contains an optional (default = off) 16MBit SPI flash chip that is controlled by FeatherLib.

In order to keep the maximum number of pins available to customers, the SPI flash is disabled by default, but can be enabled with USB Mass Storage support so that you can access the contents on the flash memory from your PC to easily exchange data and files. Simply solder the **SPIFCS** solder jumper on the bottom of the device closed, and make sure you are running FeatherLib version 0.6.0 or higher to enabled flash and USB mass storage support.

The 16MBit SPI Flash is enabled starting with FeatherLib 0.6.0. Please make sure you are running a recent version of FeatherLib when working with the external flash memory.
The SPI3 bus used for SPI flash is controlled by FeatherLib, and the four pins shown below should be avoided in your own sketches when SPI Flash is enabled in a future FeatherLib release.



SPI flash is disabled by default. It can be enabled by soldering the **SPIFCS (A4)** solder jumper on the back of the PCB closed before powering the board up, which will connect the CS/SSEL of the SPI flash to pin **A4**:



PWM Outputs

Pins that can be used as PWM outputs are marked with a tilde character ('~') on the silk screen.

The timers associated with specific PWM outputs are listed below. These timers are important since all PWM outputs on the same HW timer will use the same period or pulse width. This means that if you set the pulse width for PA1, which uses HW Timer 5, this will also set the pulse width for PA2 and PA3 which use the same timer peripheral block.

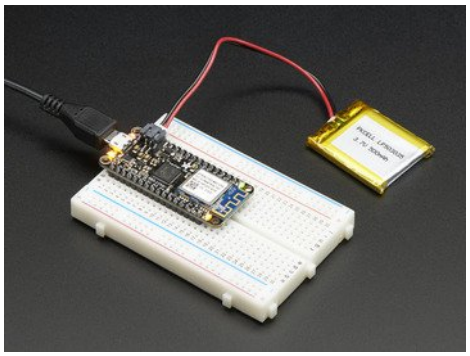
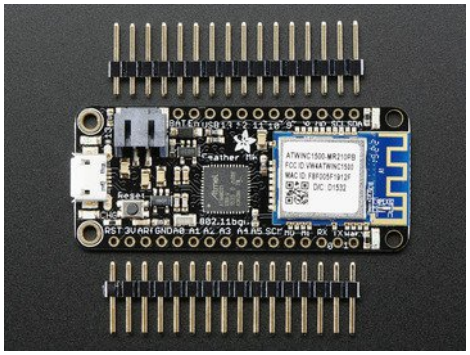
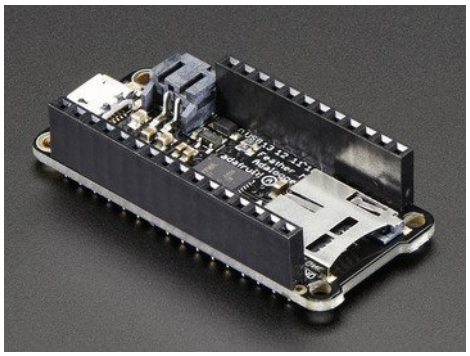
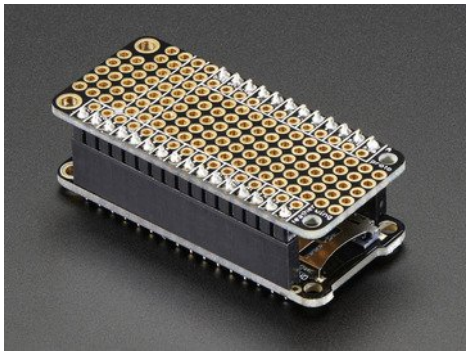
Pin Name	HW Timer	Notes
PA1	Timer 5	
PA2	Timer 5	
PA3	Timer 5	
PA15	Timer 2	Status LED
PB4	Timer 3	
PB5	Timer 3	
PC7	Timer 8	

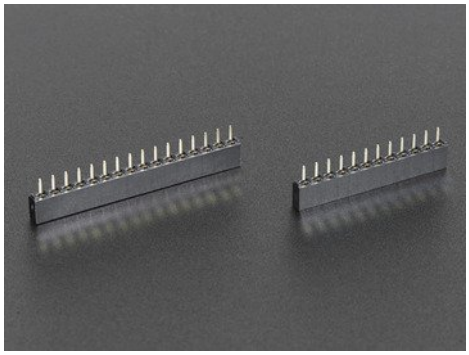
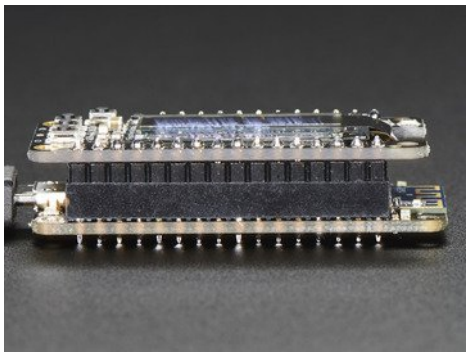
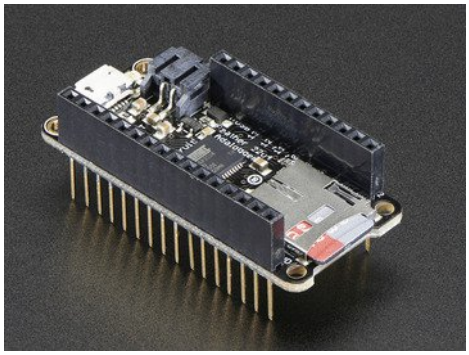
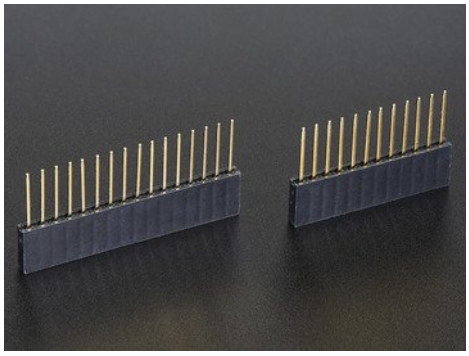
Assembly

We ship Feathers fully tested but without headers attached - this gives you the most flexibility on choosing how to use and configure your Feather

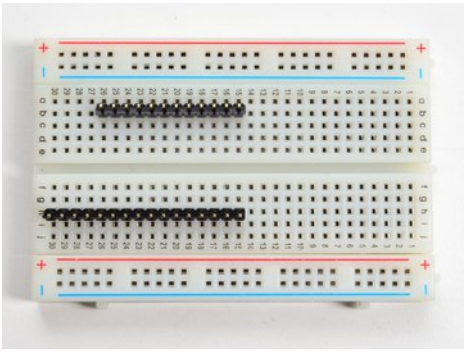
Header Options!

Before you go gung-ho on soldering, there's a few options to consider!

- 
- 
  The first option is soldering in plain male headers, this lets you plug in the Feather into a solderless breadboard
- 
  Another option is to go with socket female headers. This won't let you plug the Feather into a breadboard but it will let you attach featherwings very easily
- 

- 
We also have 'slim' versions of the female headers, that are a little shorter and give a more compact shape
- 
- 
- 
Finally, there's the "Stacking Header" option. This one is sort of the best-of-both-worlds. You get the ability to plug into a solderless breadboard *and* plug a featherwing on top. But its a little bulky

Soldering in Plain Headers

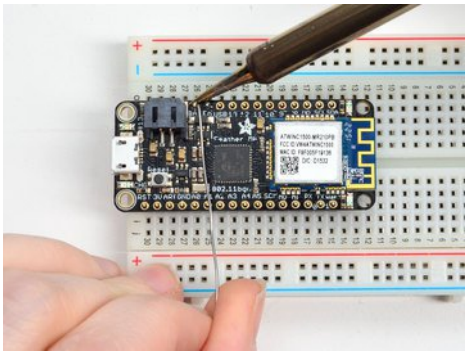
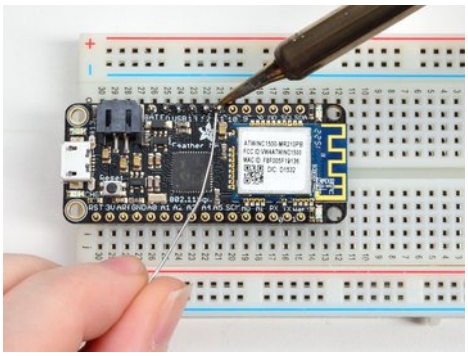
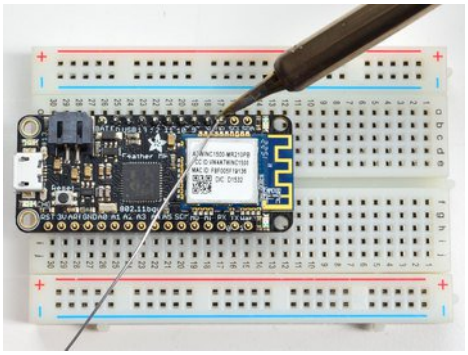
- 

Prepare the header strip:

Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**

Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout

- 
- 
- 

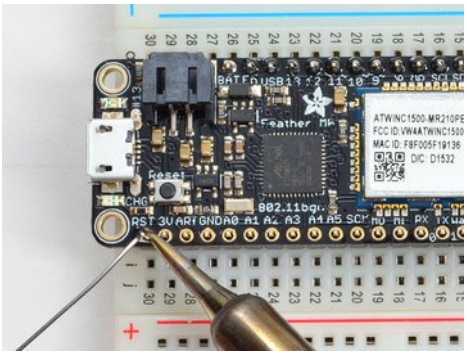
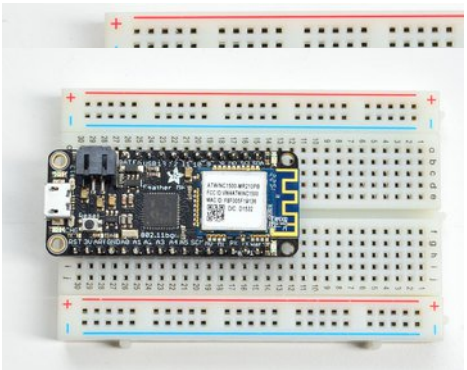
pads

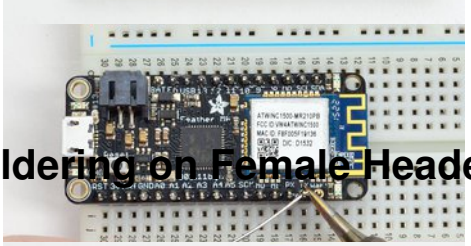
And Solder!

Be sure to solder all pins for reliable electrical contact.

(For tips on soldering, be sure to check out our[Guide to Excellent Soldering](http://adafru.it/aTk)).

Solder the other strip as well.

- 
- 

  You're done! Check your solder joints visually and continue onto the next steps
- 

Soldering on Female Header

- 

Tape In Place

For sockets you'll want to tape them in place so when you flip over the board they don't fall out