PUTSI™ WiFi Telemetry Module

Written by:
Martin Alcock, M. Sc, VE6VH

For Software Revision: 0.2

Alberta Digital Radio Communications Society,
43 Citadel Bay NW,
Calgary, AB T3G 3Y4
www.adrcs.org

Table of Contents

List of Figures

List of Tables

Document Revision History

Date	Rev	Description
June 2023	0.1	Draft of dual mode operation
Sept, 2023	0.2	Updated to current firmware
	0.2a	Revised A/D input scalars, added large Allstar configuration section
	0.2b	Moved temperature measurement to a separate document
Oct, 2023	0.2c	Rebanded with ADRCS logo
	0.2	Removed watermark added programming Appendix.

Table 1 Revision History

Additional Documents

Number	Title
AN001	Measuring Temperature

Table 2 Additional Documents

Reference Documents

[1] gnu.org, "General Public Licence," [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html. [Accessed 25th February 2018].

[2] Ettus Research, "USRP Hardware Driver and USRP Manual," [Online]. Available: https://files.ettus.com/manual/page_transport.html. [Accessed 21 October 2020].

[3] Analog Devices, "Low Voltage Temperatue Sensors," [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/TMP35_36_37.pdf. [Accessed 20 09 2023].

[4] MIcrochip technology Inc, "MPLAB® IPE: Installation Instructions," [Online]. Available: https://www.microchip.com/en-us/education/developer-help/learn-tools-software/mcu-mpu/mplab-ipe/installation. [Accessed 23 10 2023].

Glossary of Terms

PBX	Private Branch Exchange. A node in a telephone network that provides connectivity for a series of local extensions to a set of trunks.
VOIP	Voice over Internet Protocol. A system where telephone calls are placed, and audio is exchanged using the Internet Protocol.
PUTSI	PIC USB Telemetry System Interface
Wi-Fi	Wireless Fidelity implementing the IEEE 802.11x standards.
USRP	Universal Software Radio Peripheral
UDP	User Datagram Protocol
SMS	Simple messaging system
USB	Universal Serial Bus
PIC	A series of microcontrollers by MicroChip, Inc.

Intellectual Property Notice

The hardware components and all intellectual property described herein is the exclusive property of the Alberta Digital Radio Communications Society and others (“the owner”), all rights are reserved.

The owner grants licence to any Amateur for personal or club use, on an as is and where is basis under the condition that its use is for non-commercial activities only, all other usages are strictly prohibited. Terms and conditions are governed by the GNU public licence [1].

No warranty, either express or implied or transfer of rights is granted in this licence and the owner is not liable for any outcome whatsoever arising from such usage.

Disclaimer

This document is a preliminary release for a product still in development and may be subject to change in future revisions. The software may be subject to unpredictable behaviour without notice. You are advised to keep a can of RAID™ Ant, Roach and Program Bug killer handy. Spray liberally on the affected area when needed.

Release notes

Initial Release version 0.1:

1. **Dual operating modes**
   The current firmware supports operation in both the Wi-Fi and USB modes, which can be used simultaneously.

2. **Bidirectional Digital I/O**
   Digital I/O is now supported in both directions in both communication modes.

3. **Bi-Colour Communications LED**
   The COM led can now be off, red, green or alternating to reflect the current state of the firmware.

4. **Wi-Fi Configuration Mode**
   The Wi-Fi parameters can be entered using a host mode and single web page.

Update Rev 0.1a:

1. **Bugs Fixed**
   A bug was found in the configuration mode that disabled both Wi-Fi and USB modes. The configuration mode is entered automatically if this condition occurs.

2. **Wi-Fi setup shortened**
   The port numbers and auto configuration were removed from the Wi-Fi setup and implemented as compilation options. Port numbers are unlikely to change, and the auto configuration is only applicable to the Wi-Fi mode, so now it is always enabled.

Update Rev 0.2:

1. **Bugs Fixed**
   Fixed bugs in ADC message and processing of Pin state messages. Input voltages were tested with Allstar, as well as alarms and output messages.

2. **Restart code added**
   Added code to the ID request message to perform a soft restart, in attempt to autonomously keep up with Allstar restarts.

Overview

The digital telemetry system is designed to gather data at a remote site by a reporting device which sends it over an IP network to a remote site, where it is written to a database to be displayed at will. The data content can be from an analog source using and off the shelf converter, or from a digital input such as an open door sensor. A mechanism is also in place to return data for digital outputs, which can be used to activate remote devices such as lights, switches, etc.

Packets are sent over an IP network using the USRP (Universal Software Radio Peripheral) [2] protocol, as user datagram packets (UDP), which does not require a connection to be maintained. Packets are acknowledged by the receiver by sending back an acknowledgement packet.

Two types of reporting devices for gathering data have been developed, the MaplePi FPGA controller and a standalone device which employs a PIC processor called PUTSI (PIC USB Telemetry System Interface) which can be used in conjunction with another device such as a Raspberry Pi or other computer over a USB connection, or it can transmit packets directly over a Wi-Fi connection.

The MaplePi FPGA controller has a driver for Allstar that has the telemetry functionality built in, which makes use of the on-chip 8 channel A/D converter, and with an additional module can implement up to 8 outputs and 3 inputs. The PUTSI is equipped with 8 analog inputs, and 5 digital inputs and outputs, and can implement the Chameleon functionality required by Allstar, or be used in a standalone mode.

The receiving site has a web-based interface and runs on a standalone server. Data from the reporting device is written to a database, which dynamically updates a web page when needed. Analog data is interpreted by a pre- and post-scalar formula into more ‘meaningful’ data, such as temperature in degrees Centigrade, Kelvin or Fahrenheit, and digital I/O can be represented as ‘real world’ devices.

Alarms can be generated that can be sent to key personnel using an external e-mail facility, or, if present, sent as a SIP message to an IP phone, or as an SMS message to a cell phone. The frequency of updates is completely programmable.

Hardware

Figure 1 illustrates the PUTSI (PIC USB Telemetry System Interface) standalone hardware module for telemetry gathering. It has a standalone processor and can be connected to an Allstar system with USB or directly to a hub using a Wi-Fi module. In the USB mode, it emulates the Chameleon device supported natively by Allstar.

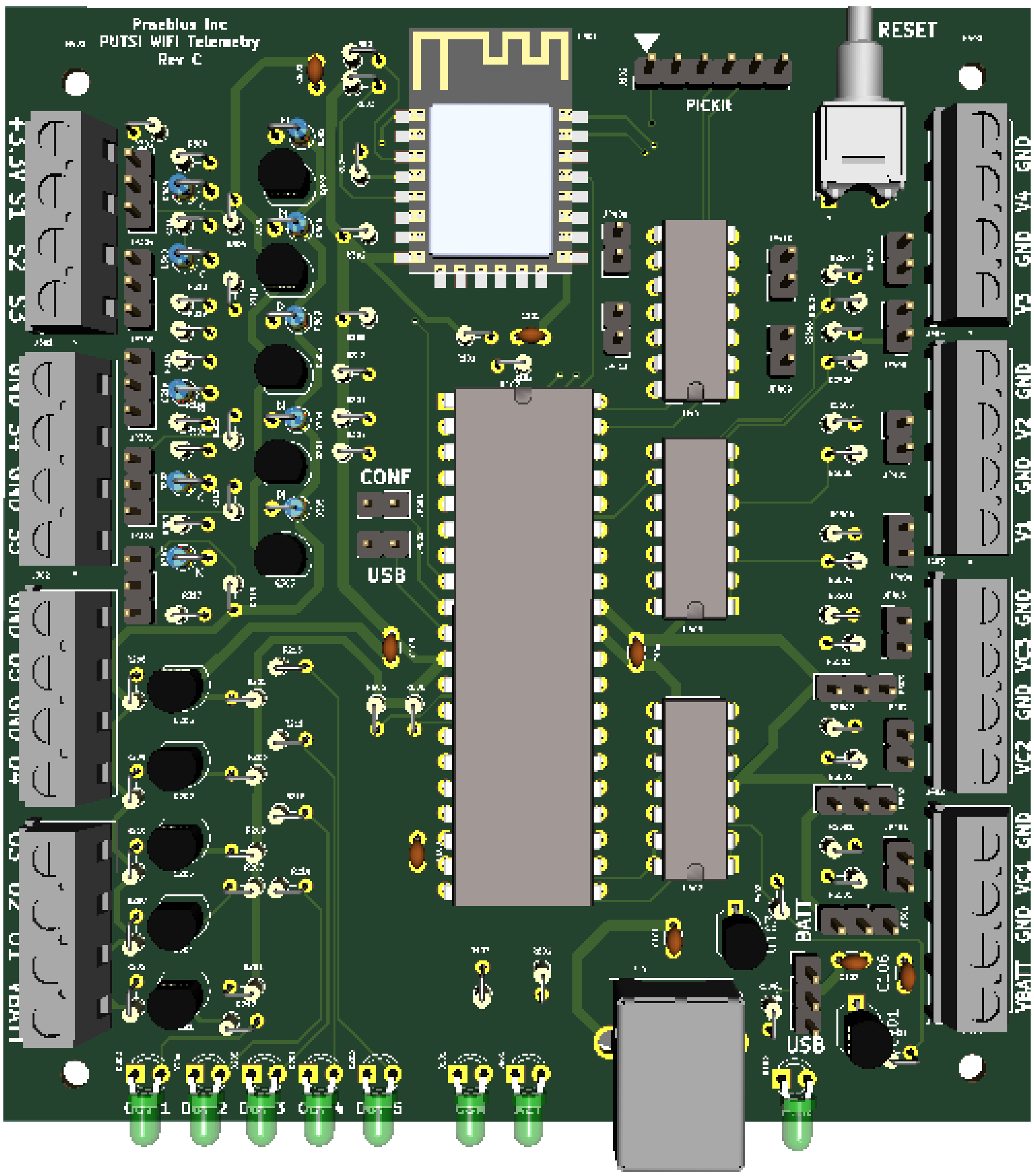

Figure 1 Pic based USB/Wi-Fi Telemetry Board (Rev C)

Specifications

The module can be powered either by the USB connection or a separate battery connection. The specifications are illustrated in Table 3.

Parameter	Min	Typ	Max	Units
Number of Analog Voltage inputs		4		
Number of Configurable Analog Channels		3		
Number of battery monitoring channels		1		
Number of uncommitted op amps		4		
Analog input voltage range	0		16[1]	V
Number of digital inputs		5		
Digital Input voltage range	0		vBatt	V
Number of digital outputs		5		
Output voltage range	0		vBatt	V
Output sink current			20	mA
Battery input voltage (vBatt)	5	12	16	V

Table 3 PUTSI module specifications

Power Options

The board can be powered either from the +5V supply on the USB cable, or from the VBATT terminals. Jumper JP101 determines which source is to be used, as illustrated in Table 4.

JP101 Jumper Settings	Power Source
1-2	+5V from the USB cable (recommended)
2-3	From the VBATT terminals (J401 1-2)

Table 4 Power Options

LED Indicators

There are 8 LED indictors as shown in Table 5.

LED Name	Purpose
CONN	Indicates that a Bluetooth connection has been established
PWR	Indicates power has been applied, through either source
COM	Indicates that the board is communicating with the host computer
OUT 1 to 5	One LED to show the status of each digital output

Table 5 LED indicators

[1] Higher voltages can be accommodated by adjusting Rg and Rs for the specific input.

Reset

There is a reset toggle switch beside the COM LED. Depressing it will reset the PIC processor, to re-establish communications the host software may need to be restarted.

Connectors

There are 8 connectors for inputs and outputs:

Connector	Pin	Desig	Purpose
J401	1	VBATT	Battery connection for power or monitoring
	2	GND	
	3	VC1	Configurable voltage input #1
	4	GND	
J402	1	VC2	Configurable voltage input #2
	2	GND	
	3	VC3	Configurable voltage input #3
	4	GND	
J403	1	V1	Analog input #1
	2	GND	
	3	V2	Analog input #2
	4	GND	
J404	1	V3	Analog input #3
	2	GND	
	3	V4	Analog input #4
	4	GND	
J201	1	VBATT	Battery voltage output
	2	O1	Digital Output #1
	3	O2	Digital Output #2
	4	O3	Digital Output #3
J202	1	O4	Digital Output #4
	2	GND	
	3	O5	Digital Output #5
	4	GND	
J302	1	S5	Sense Input #5
	2	GND	
	3	S4	Sense Input #4
	4	GND	
J301	1	S3	Sense Input #3
	2	S2	Sense Input #2
	3	S1	Sense Input #1
	4	VCC	Regulated +3.3V output

Table 6 Connector Pinouts

Converter Range

The A/D converter has a resolution of 3.3V/1024 per count. Using a resistor divider of 820/3300, the battery voltage can be scaled to fit into its linear range as shown in Figure 2.

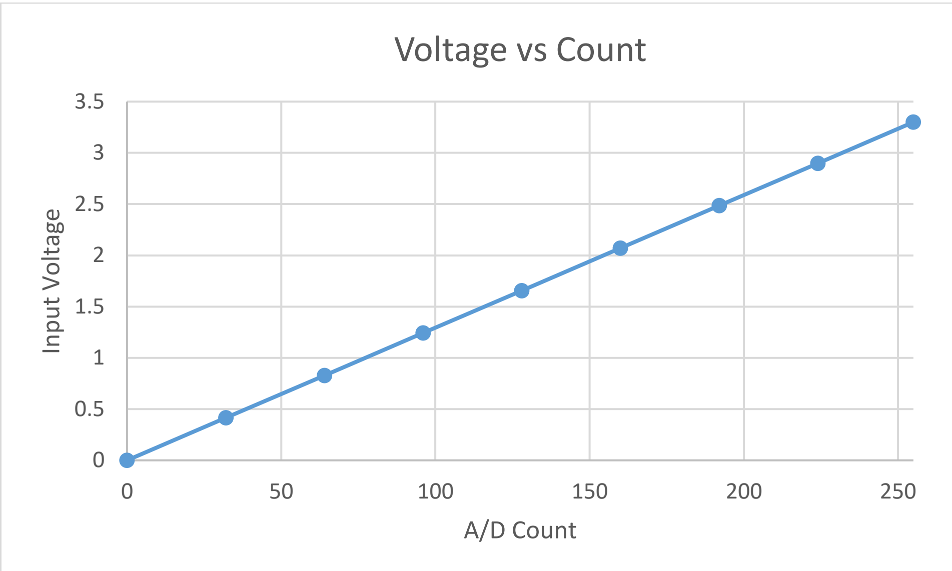

Figure 2 Input voltage vs A/D count.

Analog Inputs

At each input a dedicated op amp follower provides buffering before the converter. Figure 3 illustrates one of the analog voltage inputs:

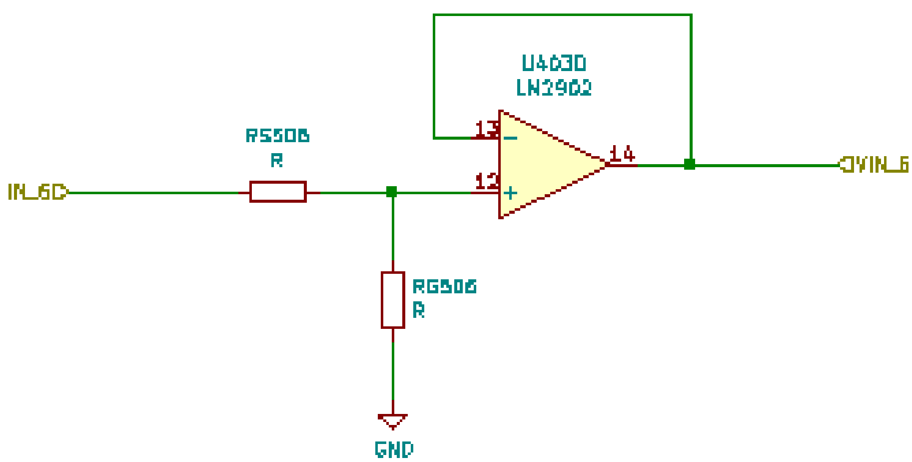

Figure 3 Analog input

Choosing the Resistor Divider Values

Resistors RS and RG (Series and Ground) implement a voltage divider to bring the analog voltage into operating range of the A/D converter, which can be from 0 to 3.3V.

The resistor values will vary depending on the maximum input value. It is recommended that the input voltage to the A/D not exceed 3.3V, so the divider value determines how the input value is scaled. In the monitoring software, a scalar value can be applied to the received value to restore the original voltage.

Table 7 lists recommended values based on a maximum input voltage. The scalar values shown can be used to restore the original voltage in the monitoring software.

Max Input Voltage	RS (Ω)	RG (Ω)	Max A/D Input	A/D max Count	Allstar Scalar	Scaled
24	3300	470	2.99	232	9.67	24
16	3300	820	3.18	247	15.44	16
14[2]	3300	910	3.03	234	16.71	14
8	3300	2200	3.20	248	31.00	8
6	3300	2900	2.81	217	36.17	6

Table 7 Recommended Resistor Values

The formula for calculating the resistor values is shown in the following equation:

$$3.3V = Vin \times \frac{Rs}{Rs + Rg}$$

For example, if a battery voltage was to be measured that had a maximum of 16V, the Rs can be calculated by first fixing Rg at a known value, and then solving for it. The recommended values of 820/3.3K will yield an approximate divide by five, which will work for most voltages used in repeater systems.

Each input has a jumper before the resistive divider. There are four uncommitted op amps configured as unity followers, and the jumper terminals can be connected to one of these to provide high impedance isolation of any input, before the divider.

[2] Default value for initial setup

Configurable Analog Inputs

Configurable inputs have a different circuit as illustrated in Figure 4. There is an additional jumper that enables the RG resistor to either act as a divider, when connected to ground, or as a voltage source when connected to Vcc.

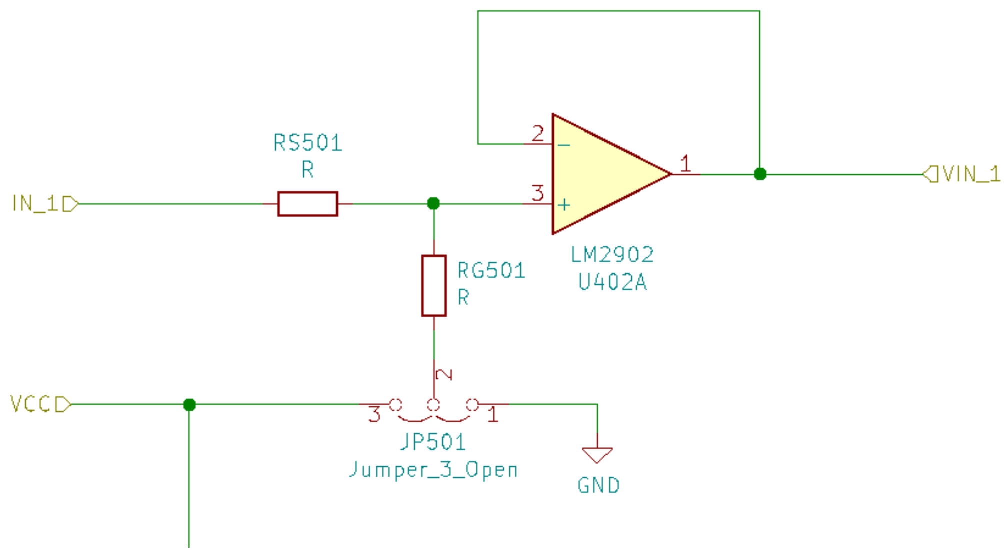

Figure 4 Configurable analog Input

A thermistor type temperature probe or active device can be configured for measuring temperatures, such as the shack, power amplifiers, etc. The recommended devices for temperature measurement is the TMP36 from Analog Devices [3], which retails for about $3.

For additional information, please consult document AN001: Measuring temperature.

Digital Inputs

Digital inputs can sense the presence of an analog voltage up to 12v, which can be hooked up to sense environmental changes such as a door being opened. A typical input circuit for each digital input is shown in Figure 5.

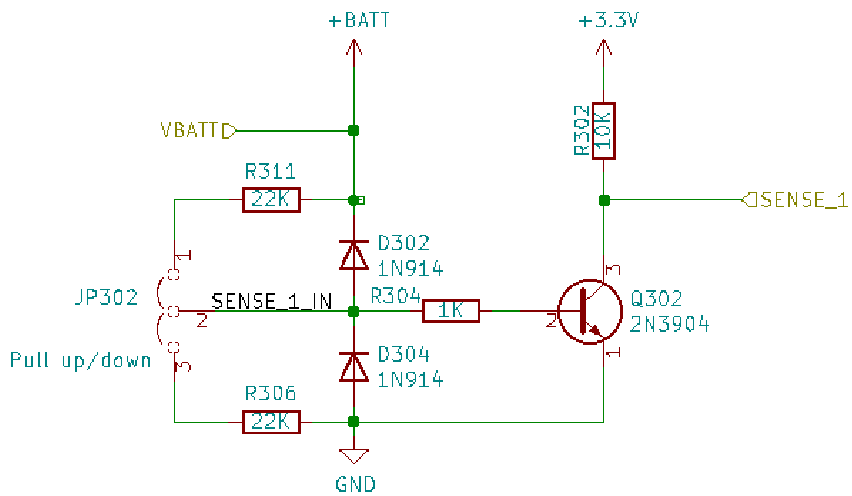

Figure 5 Digital input conditioning

Each input can be pulled up or down depending on the jumper setting. Following that a pair of protection diodes ensure that the voltage does not go below ground and above the battery voltage. It this were to occur it is clamped. The transistor buffers and inverts the input, the software removes this inversion. The quiescent voltage when pulled up is about 1V, it can be changed by modifying the pullup resistor value.

Digital Outputs

The circuit of a digital output is shown in Figure 6. Each has a light emitting diode (LED) to show its status and is buffered with a field effect transistor (FET). When the output from the microprocessor is at ground, the LED is turned off and no current is sourced to the gate of the FET, and the output rises to the battery voltage. When current is applied by the microprocessor, the LED is lit up and the FET is turned on.

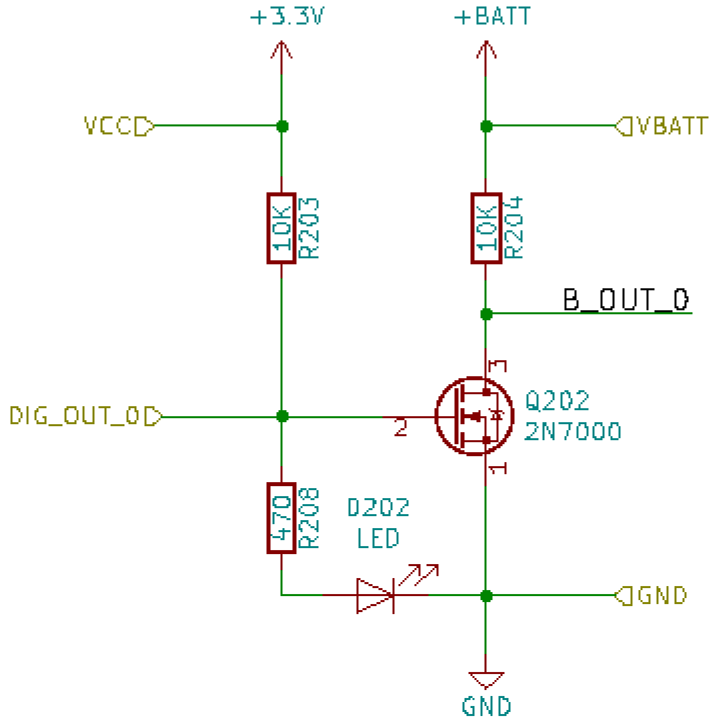

Figure 6 Typical Digital Output

Configuring PUTSI

Configuring for USB mode only

The device is preconfigured to work in the USB mode with the Wi-Fi disabled. No further configuration is required if this is the only mode of operation required. All setup parameters are ignored as the USB requires no further configuration.

Configuring for WiFi mode

If the WiFi mode is to be used, either in lieu of the USB or in tandem with it, it has to be configured before it can be used. To enter the configuration mode, insert a jumper into the 'CONF' pins close to the processor, and toggle the reset switch. It will come up with the COM led flashing Red.

The Wi-Fi setup page can be launched by connecting to the network 'PUTSI Wi-Fi', using a PC, smart phone or tablet; there is no password needed. Launch a web browser and go to address 192.168.36.1, the device will respond with the configuration page as shown in Figure 7.

All parameters for the server IP, Radio ID, Network name and password must be entered with the correct information.

Dual mode

The device can operate in both modes simultaneously, but this has to be configured using the Wi-Fi setup page.

WIFI Setup Page

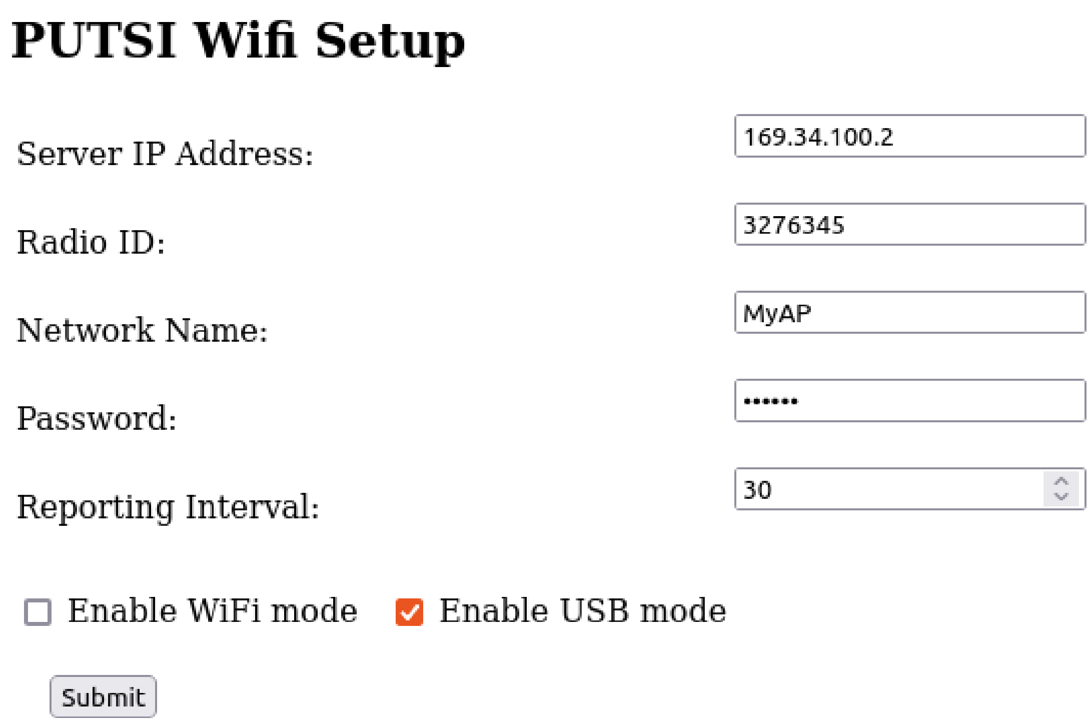

Figure 7 Wi-Fi Configuration Page

The settings are illustrated in Table 8.

Setting	Contents
Server IP Address	IP address of the telemetry server on the network for WiFi mode
Radio ID	Unique radio ID, up to 7 characters
Network Name	Name of the network where to connect (Access point)
Password	Network password
Reporting Interval	Interval between reports in seconds; recommended value is 30.
Enable WiFi Mode	Enables the transmission of packets over WiFi
Enable USB Mode	Enables the USB mode

Table 8 Wi-Fi Configuration settings

Configuring Allstar

There are four configuration entries that need to be considered, all of which are entered into the rpt.conf configuration file in /etc/asterisk. After configuration, the analog inputs can be calibrated.

1. Enter the base setup so that Allstar can find the device and know how the pins are mapped. This cannot be changed as it is fixed in PUTSI, however the configuration data is processed.
2. Enter the [meter-faces] section to correctly interpret the analog inputs.
3. Enter alarm conditions to process digital inputs.
4. Enter functions to respond to alarms and set outputs.

Base configuration

The base configuration lists all the data acquisition devices, (usually one), their hardware type and to which physical port they are connected. Following that is the pin definitions.

```
; Data Acquisition configuration
[daq-list]
device = daq-putsi

[daq-putsi]                 ; Defined in [daq-list]
hwtype = uchameleon         ; DAQ hardware type (fixed in app-rpt)
devnode = /dev/ttyACM0      ; DAQ device node
(pins go in here…)
```

Pin Definitions

Allstar maps the 8 analog inputs, 5 digital inputs and 5 digital outputs to a series of 18 contiguous pins, starting at Pin 1. The Allstar pin to actual input or output are shown in Table 9.

Allstar Pins	Type	Physical Input
1-8	inadc	Analog channels 1-8
9-13	inp	Inputs 1-5
14-18	out	Outputs 1-5

Table 9 Allstar pin mapping

All functions must reference the pin by the Allstar pin number.

The pin definitions are fixed and must be entered as shown below.

```
1 = inadc                  ; Pin definition for an ADC channel
2 = inadc
3 = inadc
4 = inadc
5 = inadc
6 = inadc
7 = inadc
8 = inadc
9 = inp                    ; Pin definition for an input
10 = inp
11 = inp
12 = inp
13 = inp
14 = out                   ; Pin definition for an output
15 = out
16 = out
17 = out
18 = out
```

Figure 8 Basic Setup

Meters

Meters translate the reading from the A/D converter into useful units such as voltage, temperature, wind speed, etc. The declarations are in the [meter-faces] stanza, which takes three parameters, a scale function, which words to say, and the calculated value. There are three parameters to the scale function, as below:

1. An offset to be added first.
2. Scale divider, full scale reading/number of whole units, division occurs second.
3. A post offset added after the division.

The words represent word files in /var/lib/asterisk/sounds, and its sub-directories. The first part is implied, the remainder of the path must be entered. The extension can be omitted.

The result of the scale function is represented by a question mark (?) as a parameter in the command.

The syntax of the meter face command is:

```
facename = scale(scalepre,scalediv,scalepost),word/?,...
```

For the battery voltage, an example syntax would be:

```
[meter-faces]
batvolts = scale(0,16.71,0),rpt/thevoltageis,?,ha/volts
```

Figure 9 Sample Meter Face for battery voltage

Where the scalar is 16.71, the default value as shown in Table 7.The words ‘thevoltageis’ and ‘volts’ exist in sub-directories of the *sounds* main directory.

Alarms

Alarms monitor input pins and execute functions based on their change of state. The declarations are made in the [alarms] stanza, which takes 7 parameters, as listed in *Table 10*.

Parameter	Purpose
name	A unique tag identifying the alarm
device	Set to [daq-putsi]
pin	Pin number to monitor, as defined in Figure 8.
ignorefirstalarm	set to 1 to throwaway first alarm event, or 0 to report it
node	the node number where the function are defined
func-lo	the function to execute on a high to low transition
func-high	the function to execute on a low to high transition

Table 10 Allstar alarm parameters

The syntax to generate an alarm from a door switch attached to the first input (Pin 9), and to monitor an AC power failure is as follows:

```
[alarms]
doorpic = daq-putsi,9,0,(your node number),*852,*851
pwrfailpic = daq-putsi,10,0,(node),*862,*861
```

Figure 10 Sample Alarm statement

When the door is opened, function 851 is executed, when closed, 852. These are explained in the Functions section.

Controlling Outputs

The digital outputs are controlled by the 'userout' function, the declarations are in the [functions] stanza. It takes four parameters:

Parameter	Purpose
Device	Device to control, set to 'daq-putsi'
Pin Number	Output pin number as defined in Figure 8.
State	1 to turn on, 0 to turn off
Playback	File to playback for an audio response

Table 11 Output function definition

An example of a function to turn on the first output is:

```
userout,daq-putsi,14,1,rpt/on
```

Functions

Functions entry into generating telemetry messages as well as controlling digital outputs. There are quite a large number of functions that are already defined; however it was discovered that any value starting with 82 through 89 were unused. The functions can be configured as shown in Table 12. The generic form of a function is:

```
[functions]
(DTMF Code)=[Statement]
```

DTMF Code	**Statement**	**Function**
841	meter,daq-putsi,1,batvolts	Invokes the meter-face as shown in Figure 9 to announce the battery voltage
851	playback,ha/dooropen	Announces that the door is opened in response to an alarm on Pin 9.
852	playback,ha/doorclosed	Announces that the door has been closed.
861	playback,ha/dcoff	Announces that the AC power is off
862	playback,ha/dcon	Announces that the AC power has been restored
870	userout,daq-putsi,14,1,rpt/on	Turns on Pin 14 (output 1) and announces it
871	userout,daq-putsi,14,0,rpt/off	Turns off Pin 14 (output 1)
872-877	**(repeated for pins 15-18)**	
878	userout,daq-putsi,18,1,rpt/on	Turns on Pin 18 (output 5)
879	userout,daq-putsi,18,0,rpt/on	Turns off Pin 18 (Output 5)

Table 12 Function definitions

Calibration

Due to component tolerances, the actual voltage returned may not be exact. To calibrate it, start with the recommended scalar values, then a calibrated value can be calculated. Connect a voltage source to the battery input, preferably at 10.0V as it makes the calculation easy. If that is not available, then measure the voltage at the input terminals.

Start Allstar and wait for it to stabilize and execute the DTMF function (see Table 12) to announce the voltage reading. A new calibration factor can be calculated from the voltage reading using the formula:

$$New\ Scalar = \frac{Scalar\ x\ Reading}{Input\ Voltage}$$

The meter faces scalar can be updated with the new scalar value, then restart asterisk to accept the new values.

Operation

USB Mode

The USB interface provides a popular connection to most devices. It follows the protocol of the USB chameleon device that was initially supported by AllStarLink. All data is sent as ASCII characters and is terminated with a line feed ($0A_{16}$) character.

There are three phases of communication:

1. Handshake
2. Configuration
3. Data Exchange

The handshake mode is entered after powering up and when the reset button is depressed. The COM LED is off until the handshake has been completed. The device waits for a message from the host to send an identification string, and, if recognized, the host then sends a command to activate the COM led, which be on a solid RED when the connection has been established. A response is sent to the host at least once every 50ms to maintain the connection. If no data has been solicited or is ready to be sent, an empty line is sent to maintain the connection.

There are two types of data exchanges, host originated and asynchronous. The host originated mode is used to solicit data from the analog channels as well as changing the digital outputs. When the hosts wants to get an update from the analog inputs, it sends the channel number and type information, and the device responds with the current reading, in the range of 00_{16}-FF_{16}.

For digital outputs, the channel number and state are sent in the message, no response is sent. A change in a digital input results in a message being sent asynchronously to the host with no solicitation.

Asterisk (Allstar) only performs the handshake and configuration once after it has been restarted. If the communication fails, the USB mode will be disabled after 2 seconds of inactivity and the COM Led turned off. To restart it requires that asterisk also be restarted. From the console, as a super-user, it can be accomplished with the command:

```
# service asterisk restart
```

WiFi Mode

Once the client data has been configured and, if inserted, the CONF jumper is removed, the reset key is depressed to restart the device. If a connection to the network is successful, the COM led is set to a steady GREEN. The ACT led shows when data traffic is being over the wireless connection.

Periodic updates are sent to the monitoring server based on the interval specified on the configuration page, the inputs and outputs are only scanned once during this interval, responses to a digital output command can be delayed by this time as well.

If both modes are enabled, then the COM Led will alternately flash red and green when BOTH connections are established.

COM Led

The COM Led is a bicolour LED that can be red or green, depending on the operational mode. The LED can either be on or flashing, as illustrated in Table 13.

Color	Mode	Meaning
NONE	Off	No communication mode is currently active.
RED	Flashing	Configuration mode has been entered; AP is active.
RED	Steady	A connection has been made to a WiFi access point; the USB mode has not completed a handshake.
GREEN	Steady	The USB is connected to a host computer and the handshake is compete. No Wifi Connection has been made.
RED/GREEN	Flashing	Both USB and WiFi modes are operational.

Table 13 COM Led modes

Appendix A: WiFi Telemetry Packet Specification

Telemetry packets use the USRP protocol format and are sent using the User Datagram Protocol (UDP) for its transport, which is connectionless. Each packet consists of several fields as illustrated below:

USRP Header 32 Bytes	TLV Header 3 Bytes	Payload Data Up to 255 bytes

Table 14 Telemetry packet format

USRP header

The packet header fields are shown in Table 15.

Field Name	Size (Bytes)	Type	Value	Usage for Telemetry
Eye	4	byte	'USRP' in ASCII	Required
Sequence	4	unsigned integer	Sequencer counter	Incremented by one
Memory	4		Memory ID or zero	Ignored
Keyup	4		1 to indicate transmitter keyed, 0 otherwise	
Talkgroup ID	4		ID of the current talkgroup	
Type	4		Packet type, see Table 16	only TLV used for telemetry
MPXID	4		Multiplex ID	ignored
Reserved	4		Reserved for future use	

Table 15 USRP Header fields

There are 7 defined packet payload types, all telemetry packets are TLV packets. The others are listed purely for documentation purposes.

Packet Type	Definition	Contents
0	Voice	Voice coded as 16-bit linear PCM in little endian format
1	DTMF	DTMF coded as per RFC4733
2	TEXT	Text message
3	PING	Ping message
4	TLV	Type-Length-Value packet
5	ADPCM	Voice coded as adaptive delta pulse code modulation
6	ULAW	Voice coded using G711 µLaw

Table 16 USRP packet types

TLV Header

The TLV header is defined in Table 17.

Byte	Size	Content
1	8-bit unsigned integer	Type field
2		Length of entire payload (including TLV header)
3		Value field, up to 255 bytes

Table 17 TLV packet header

There are 13 different types of TLV payload types as illustrated in Table 18, all telemetry packets use a type 12, the others are listed for documentation purposes.

Type	Payload
0	Begin transmit tag
1	AMBE coded voice packet
2	End transmit tag
3	Tune
4	Play AMBE packet
5	Remote Command
6	AMBE_49
7	AMBE_72
8	Information packet
9	IMBE
10	DSAMBE
11	File transfer
12	Telemetry

Table 18 TLV packet types

The telemetry system uses a type 12 packet.

Payload Data

The payload data contains a radio ID, and a series of data fields, up to the maximum data field size, as shown in Table 19.

Radio ID	Data Field 1	Data Field 2	Data Field n

Table 19 Payload Data fields

Data fields contain three different entries as illustrated in Table 20

Entry	Size (bytes)	Contents
Entry type	1	Identifies the entry type
Number of Entries	1	Number of entries in the field
Field Data	1-2	Field data in big endian format

Table 20 Information packet fields

There are three entry types as shown in Table 21:

Type	Size (bytes)	Data type
0	2	Analog input reading
1	1	Digital inputs, zero=off, non-zero=on
2	1	Digital outputs

Table 21 Entry types

Packet Exchange

The frequency of packets is up to the telemetry device to determine. At the receiver the database is updated immediately, the display is dynamically updated each time a new record is written to the database.

The uploaded packet must contain all field types, all fields must reflect the current state of all inputs and outputs. A response packet is sent back immediately, and if the device supports digital outputs this field will be included, otherwise the payload field will be empty. The response packet will contain the same sequence number in the header as the uploaded packet, so this field can be used by the telemetry device as an implicit acknowledgement.

If no response is heard, then it can be assumed that the link is down, and the packet was not received. The recovery procedure is up to the specific device. It is recommended that local statistics be kept for this purpose.

Appendix B: Programming the MPU

Requirements

To program the MPU, you will need the following:

1. A copy of the Microchip IPE™ programming code [4]. The URL is in the reference section.
2. A completed board with the microcontroller inserted.
3. A in-circuit probe such as a PicKit™.

Setting up the Programming Application

Setting up the application involves three steps:

1. Selecting the part to be programmed.
2. Connecting to the programming probe.
3. Selecting the file to be programmed.

After that, one or many parts can be programmed.

Selecting the part to be programmed
First, launch the IPE application, and click the 'Operate' Tab. The first two steps can be accomplished from the 'Device and Tool Selection' frame in the dialog box. Select 'Advanced 8-bit MCU's (PIC18) from the 'Family' drop down box, and PIC18F45K50 from the 'Device' drop down, as illustrated in Figure 11. When done, click the 'Apply' button.

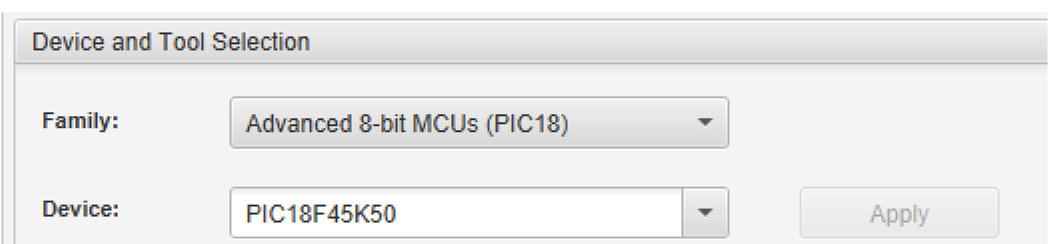

Figure 11 Part Selection

Connect to the Programming Probe

Connect the programming probe to the PUTSI board, note the pin 1 orientation marked on the probe and the board with a triangle. The probe should connect directly to the board without a cable. Figure 12 shows the marking for pin 1 on the probe, Figure 13 shows the same for the board. Note that it should face away from the board components for the proper orientation.

Figure 12 Pin 1 orientation on probe

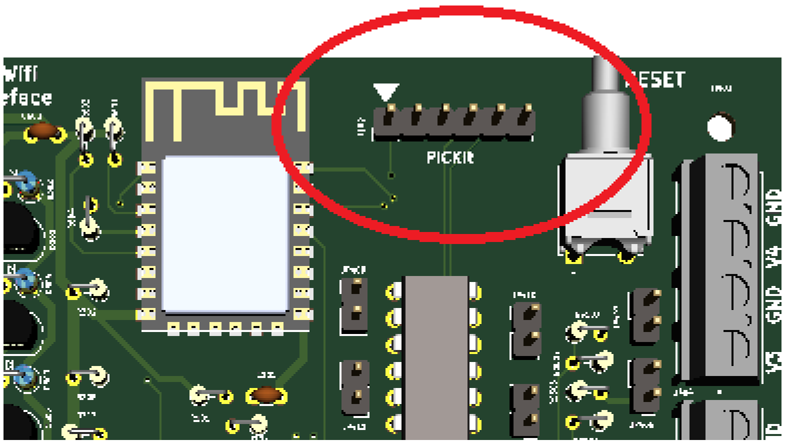

Figure 13 Pin 1 orientation on board

Power the board and ensure that the power LED is lit. Select the tool from the 'Tool' drop down list, and click the 'Connect' button as show in Figure 14.

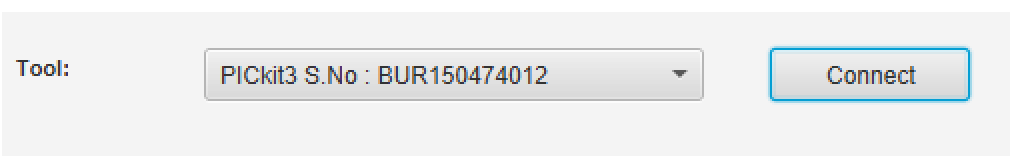

Figure 14 Tools selection drop down list

If the output (log) window is being shown, click the 'Operate' tab to return to the setup.

Note: if you are using a PICKit 5, there are 8 pins on the programmer. Align the triangles for pin 1 as shown, and the last two pins will not be connected. Open the 'Advanced' Tab on the programming software and ensure that the pull-ups or pull-downs on PGC and PGD are set to 'None'. Also enable the slow programming mode.

Selecting the file to be programmed

Download the latest .hex file from the Github site 'Code' directory. The current version is called:

PUTSI_Wifi.V02.hex

Click the 'browse' button beside the 'Hex File' entry in the frame below the Deice and Tool selection as shown in Figure 15.

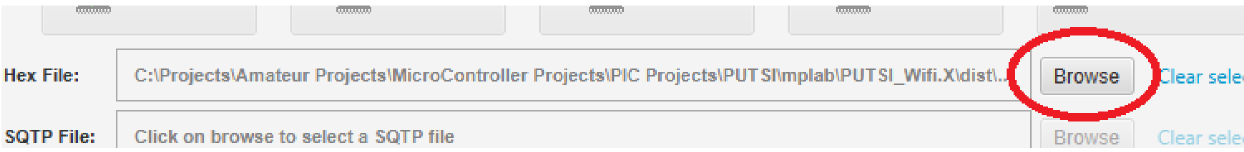

Figure 15 Selecting the programming file

A file selection box will appear. Navigate to where the file is located, click on the file name and then click the 'Open' button, as illustrated in

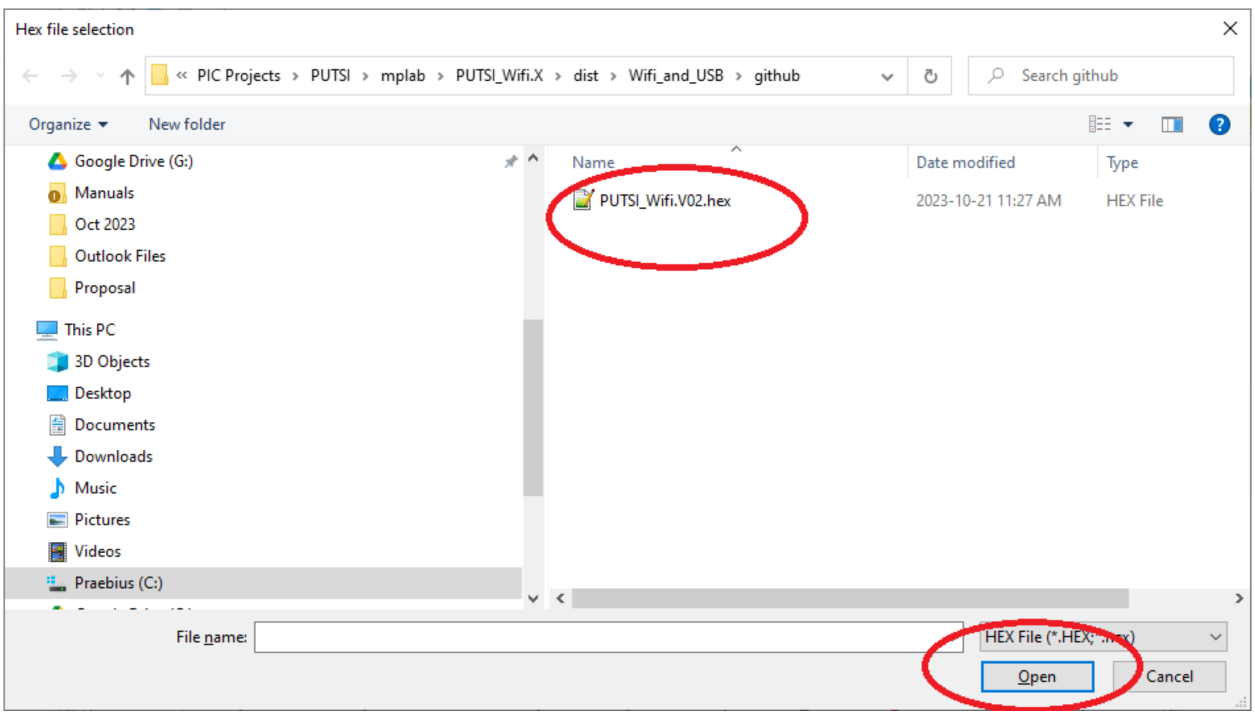

Figure 16 Selecting the file

You should see a message in the output (log) window that the file has been successfully loaded. To continue, click the 'Operate' tab again.

Programming the Part

Once all the previous steps are done, the buttons to carry out the programming should be enabled. If not, repeat the previous steps and ensure that the probe is properly connected, and the board is powered up.

Finally, click the 'Program' button as shown in Figure 17.

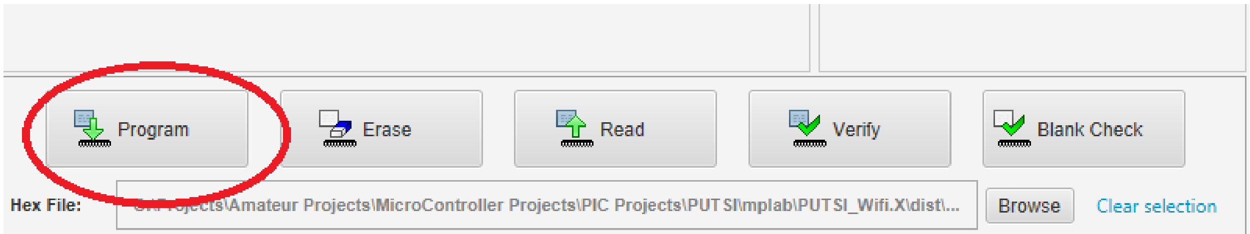

Figure 17 Programming the part

When completed, the probe can be removed from the board, while still powered. The board is now ready for use.

If you want to program another part in the same board, the board can be powered down with the probe in place, and the part changed. The program will complain that the target has been removed, however it will recover when the board is powered up again. Repeat this step to program the next part.