USER GUIDE

E20

SNAP Enabled Gateway
Version 1.1 for
Firmware Versions 1.X and Higher

©2015 Synapse, All Rights Reserved. All Synapse products are patent pending. Synapse, the Synapse logo, SNAP, and Portal are all registered trademarks of Synapse Wireless, Inc.

Doc# 116-031520-002-C000

6723 Odyssey Drive // Huntsville, AL 35806 // (877) 982-7888 // Synapse-Wireless.com

Disclaimers

Information contained in this document is provided in connection with Synapse products and services and is intended solely to assist its customers. Synapse reserves the right to make changes at any time and without notice. Synapse assumes no liability whatsoever for the contents of this Manual or the redistribution as permitted by the foregoing Limited License. The terms and conditions governing the sale or use of Synapse products is expressly contained in the Synapse's Terms and Condition for the sale of those respective products.

Synapse retains the right to make changes to any product specification at any time without notice or liability to prior users, contributors, or recipients of redistributed versions of this Manual. Errata should be checked on any product referenced.

Synapse and the Synapse logo are registered trademarks of Synapse. All other trademarks are the property of their owners. For further information on any Synapse product or service, including technical assistance, contact us at:

Synapse Wireless, Inc.
6723 Odyssey Drive
Huntsville, Alabama 35806
256-852-7888
877-982-7888
256-924-7398 (fax)

www.synapse-wireless.com

License governing any code samples presented in this user guide

Redistribution of code and use in source and binary forms, with or without modification, are permitted provided that it retains the copyright notice, operates only on SNAP® networks, and the paragraphs below in the documentation and/or other materials are provided with the distribution:

Copyright 2008-2015, Synapse Wireless Inc., All rights Reserved.

Neither the name of Synapse nor the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided "AS IS," without a warranty of any kind. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SYNAPSE AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SYNAPSE OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS SOFTWARE, EVEN IF SYNAPSE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

NOTE: This equipment is meant to be installed in accordance with all applicable local, state, and national electrical codes. If the equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired. Read these instructions, keep these instructions, and heed all warnings.

Instructions

This symbol is intended to alert the user to the presence of important operating and maintenance (servicing) instructions in the literature accompanying the appliance.

1. Overview

The SNAP Connect E20 combines a Synapse SM220 RF module and an embedded Linux-based computer to provide connectivity (Ethernet, cellular, Wi-Fi, serial,) and site aggregation capabilities to a diverse array of SNAP IoT networks across industrial temperature ranges.

TCP/IP connections can even bridge remote SNAP devices into one common network, an effective method for centralizing data storage, performing web-based analytics, and monitoring remote applications.

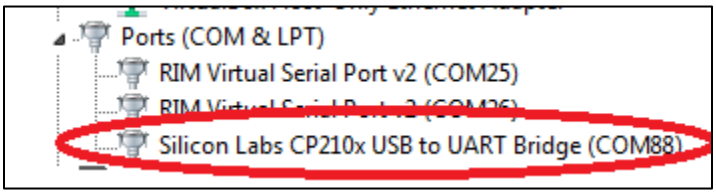
Powered by Freescale's i.MX 6 processor, the E20 has ample computing power, reliable connectivity options, and a sturdy design that makes it the ideal networking gateway for large-scale IoT deployments.

	E20
OS	Ubuntu 14.04 LTS, Linux kernel 3.10.17
CPU	ARM Cortex-A9, 800MHz (Freescale iMX6-S)
Flash	4GB eMMC
RAM	512M DDR3, 400MHz
Network	10/100 Ethernet, WiFi, SM220
USB host	1 type A
USB client	1 micro-USB - SiLabs CP2102
Operating Temperature	-40C - 70C* *UL certified for 65C maximum operating temperature*
Board Size	15.5cm x 9cm x 2cm
Input Voltage	11-26V DC *AC power supply sold separately.* *See synapse-wireless.com for more information.*
Cellular Option	Internal cell modem
Storage Expansion	uSD – internal
LEDs / Buttons	4 LEDs 3 Buttons

* When running an application that demands unusually intensive CPU/Memory resources at 70C, the temperature on the processor core might reach up to 90C resulting in performance degradation. For more information, see http://cache.freescale.com/files/32bit/doc/app_note/AN4579.pdf.

2. Quick Start Guide

1. Connect the Micro-USB Serial port on the E20 to your PC
2. Apply power to the E20
3. Find the serial port that has been assigned to the E20 by your computer:
    a. Windows:
        i. Look under “Ports” in Device Manager.
        ii. Look for “Silicon Labs CP210x USB to UART Bridge.”



    b. Linux:
        i. look for /dev/ttyUSB0 in the /dev/ directory:

```
synapse@synapse ~ $
synapse@synapse ~ $ ls /dev/ttyUSB*
/dev/ttyUSB0
synapse@synapse ~ $
synapse@synapse ~ $
```

        ii. The USB# may be higher if you have additional ttyUSB devices plugged in.
4. Using your preferred terminal emulator (Tera Term, PuTTY, minicom, etc.), connect to the E20 using the following serial port settings:
    a. Baud rate: 115200
    b. 8 bits
    c. No parity
    d. 1 stop bit
    e. No flow control
5. Initial login:
    a. Username: snap
    b. Password: synapse

NOTE: You must change your password on first boot

6. Obtaining SNAP Connect (optional)
    a. sudo -H pip install snapconnect -i https://update.synapse-wireless.com/pypi/

3. Basic Installation and Specifications

Mounting the E20

The E20 has a number of mounting holes along the outside flanges of the unit, any of which can be used for mounting the E20 to a solid surface.

(EN) SUITABLE FOR MOUNTING ON OR ABOVE CONCRETE OR OTHER NON-COMBUSTIBLE SURFACE ONLY

(FR) APPROPRIÉ POUR LE MONTAGE SUR OU AU-DESSUS BÉTON OU AUTRE SURFACE NON COMBUSTIBLE SEULEMENT

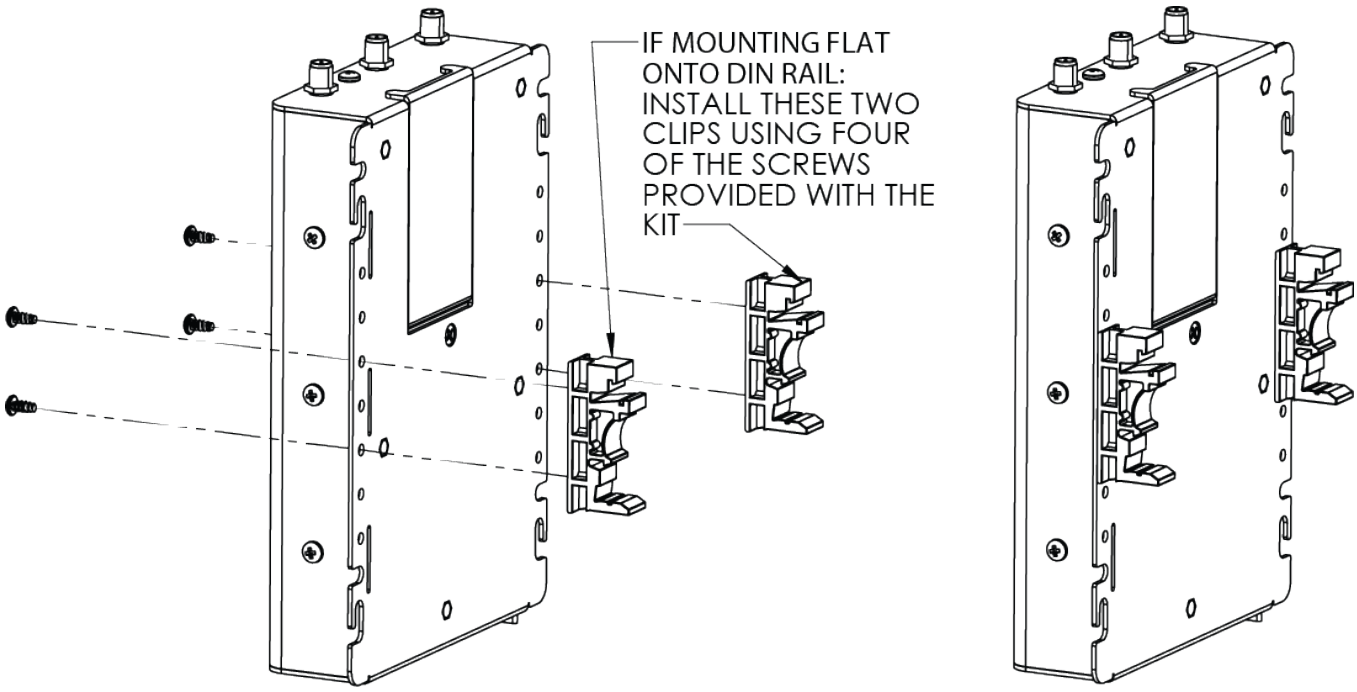
(DE) GEEIGNET FÜR DIE MONTAGE AUF ODER UBER BETON ODER ANDEREN, NICHT BRENNBAREN OBERFLÄCHE NUR

(ES) ADECUADO PARA EL MONTAJE EN O POR ENCIMA SUPERFICIE NO COMBUSTIBLE HORMIGÓN U OTRO SOLAMENTE

Synapse also provides an optional DIN Rail Mounting kit if you wish to mount the E20 on a DIN rail. (See synapse-wireless.com for part number.) There are two options for DIN rail attachment depending on your available space.

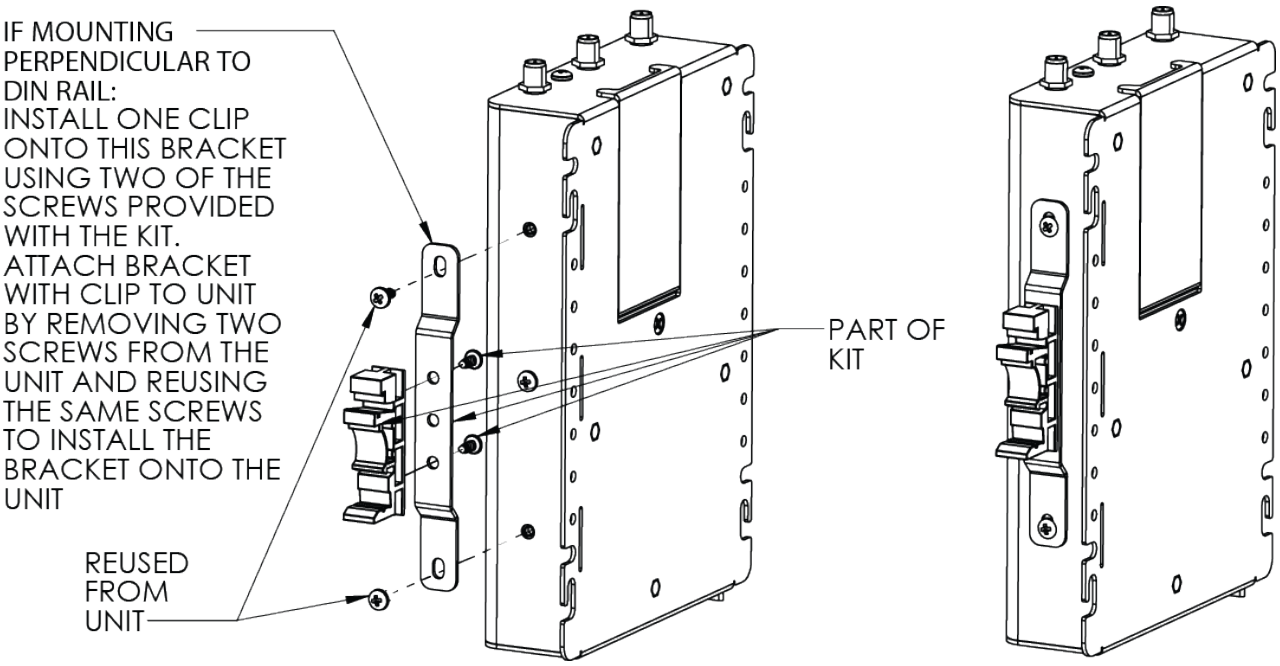
To mount the E20 flat against the DIN rail:

Attach the two white mounting clips to the E20 as shown in the diagram below:



To mount the E20 perpendicular to the DIN rail:

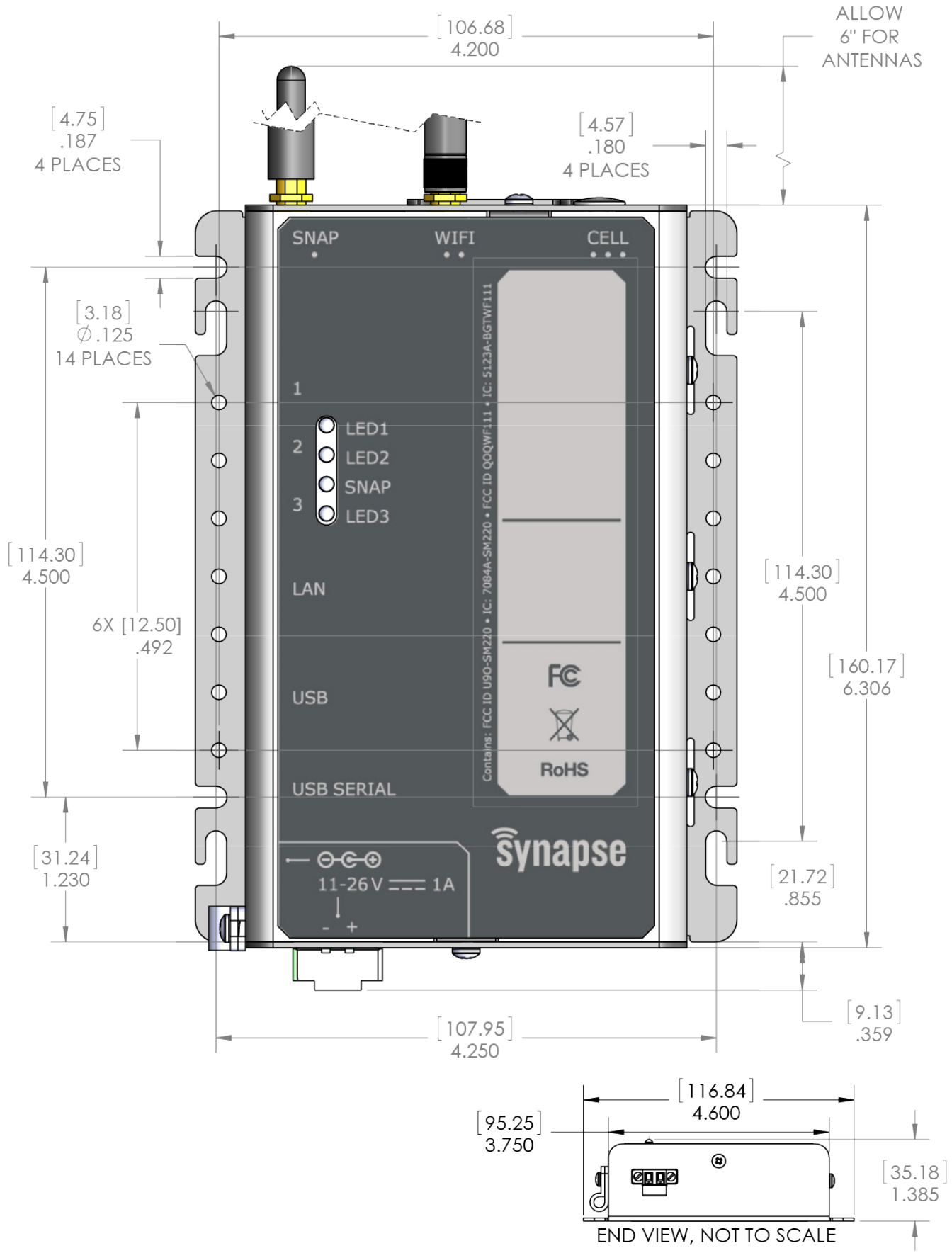
1. Attach the white mounting clip to the stand-off bracket using the two screws provided with the DIN rail kit as shown in the diagram.
2. Remove the top and bottom screws from the narrow edge of the E20, leaving the one in the center.
3. Using the screws you just removed, attach the mounting bracket to the E20 as shown in the diagram.



Connecting Power to the E20

The E20 can be powered via the barrel connector at the bottom of the left side, or the green terminal block connector on the bottom of the unit. If you're applying power via the terminal block, the positive and negative terminals are marked on the front label of the unit.

E20 Dimensions

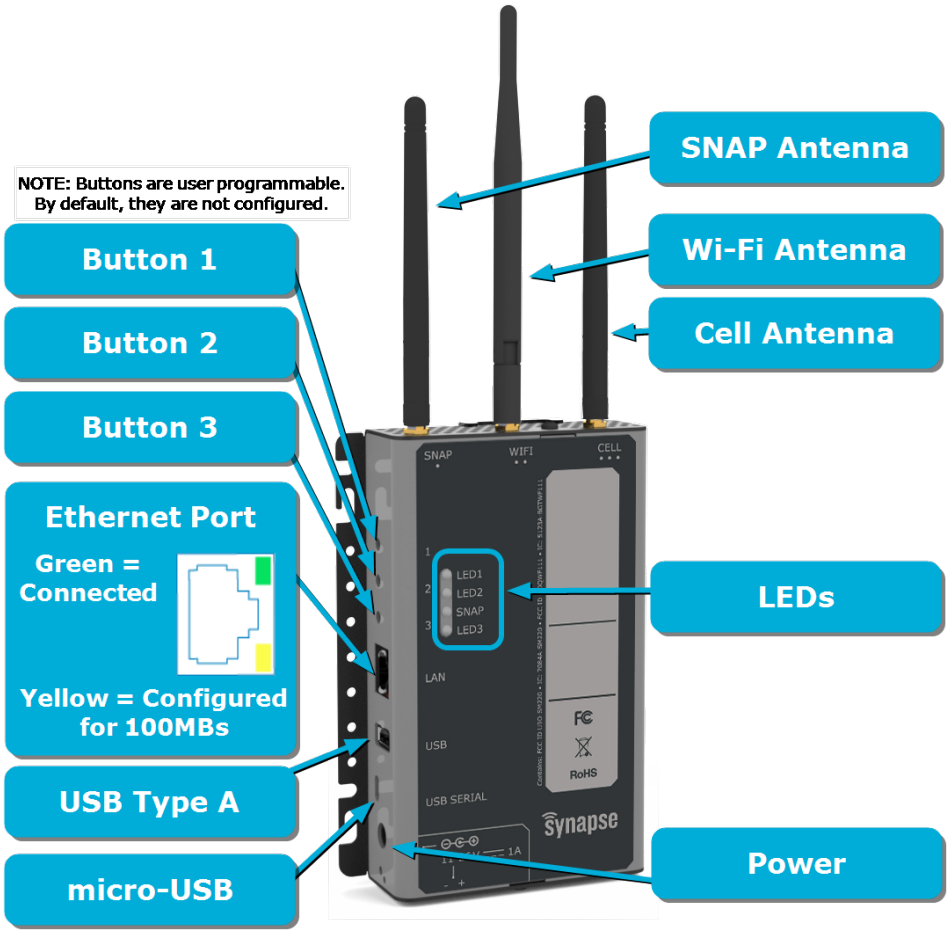


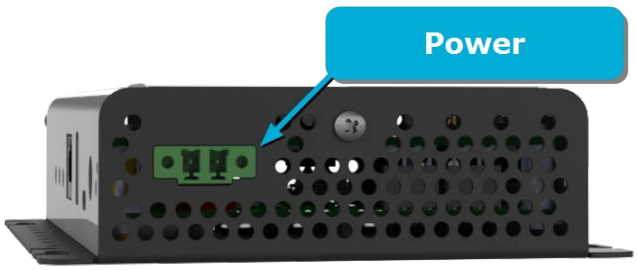
Power supply requirements:

Table 1: E20 Power

Barrel or DC Input	Min	Typ	Max	Units
Input Voltage (DC)	11		26	V
Input Current	0.13		1.0	A
USB A Port (output)				
Supply Voltage	4.75	5	5.25	V
Supply Current			500	mA

Depending on configuration, the E20 will have one to three antennas, and connectors as shown below:





4. Initial Setup

Accessing the E20 Console

The micro-USB port on the E20 provides a serial connection which can be used to access the command prompt. Additionally, the E20 is accessible via SSH over the Ethernet if you know the IP address. Generally, accessing the E20 the first time is easiest via the micro-USB port.

Initial User Settings

- Initial login credentials:
  - Username: `snap`
  - Password: `synapse`

NOTE: On first login, the password must be changed. This is a security **REQUIREMENT** so that deployed E20s will not have a known default password.

Serial Port Settings

- Baud rate: 115200
- 8 bits
- No parity
- 1 stop bit
- No flow control

Enabling Root and Changing Passwords

- Root is initially disabled. To enable you must set a root password:

```
$ sudo passwd root
```

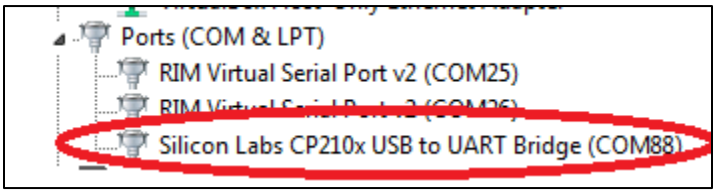
- To change the password for another username, substitute the username in the password command:

```
$ sudo passwd snap
```

NOTE: No account can connect via SSH without a password.

Connecting in Windows

- Download and install a terminal emulator such as PuTTY or TeraTerm
- If necessary, obtain the SiLabs CP210x USB to UART VCP drivers from www.silabs.com
- Connect the E20 to the computer via the micro-USB cable
- Determine which COM port your OS has allocated to the E20 serial port.
  - In Windows, look under "Ports" in Device Manager.



  - Look for "Silicon Labs CP210x USB to UART Bridge."
- Using a terminal emulator, open a serial connection to the indicated COM port

Connecting in Linux

- Download and install a terminal emulator such as minicom or screen
- Connect the E20 to the computer via the micro-USB cable
- Open /dev/ttyUSB0 in the /dev/ directory:

```
synapse@synapse ~ $
synapse@synapse ~ $ ls /dev/ttyUSB*
/dev/ttyUSB0
synapse@synapse ~ $
synapse@synapse ~ $
```

  The USB# may be higher if you have additional ttyUSB devices plugged in.
- Using your terminal emulator, open a serial connection to the indicated tty port.

5. SNAP Connect (802.15.4 Connection)

The E20 contains a Synapse SM220 module, accessible via the serial ports `/dev/snap0` and `/dev/snap1`.

SNAP Connect supplies the software infrastructure on the E20 to connect the gateway to the SNAP network via these serial ports.

To download and install SNAP Connect run the following from the E20 command line:

```
$ sudo -H pip install snapconnect -i https://update.synapse-
wireless.com/pypi/
```

After installing SNAP Connect, you'll also want to install the latest version of PyCrypto. This can be obtained via apt-get by entering:

```
$ sudo apt-get install python-crypto
```

6. E20 Specific Software Packages

A factory defaulted E20 comes with Ubuntu 14.04, running a custom 3.10.17 Linux kernel based on the imx6 kernel by Freescale. The E20 comes with several support packages pre-installed and additional ones may be installed via `apt-get` and `pip`.

NOTE: Before installing new packages, be sure to run `apt-get update` to sync your E20 with the package server, so you will obtain the newest version. The apt-get update may take a few minutes depending on your internet connection.

Additionally, it is recommended you upgrade your e20 utilities, (which are described in detail below,) by running the following command, as new versions and bug fixes may be available:

```
sudo apt-get install e20-cell-helpers e20-leds e20-gpio-scripts e20-snap-
utils
```

The E20 is pre-installed with specific `apt-get` helper packages. The packages can be found in the `/usr/local/bin` directory unless noted otherwise. They are:

e20-cell-helpers – a cell modem support package	Telit modem example scripts to reset, disable, enable the cell modem	callvz – Invokes PPPD to communicate with cell modem on Verizon power-cell-modem – Powers the cell modem enable-cell-modem – Enables the cell modem wake-cell-modem – Wakes the cell modem
	Configuration and control files for cell modems (/etc/ppp/peers)	telit-att – pppd configuration file for ATT telit-att-chat - scripted AT commands issued to modem for AT&T telit-verizon – pppd configuration file for Verizon telit-verizon-disconnect – Disconnects modem from tower verizon-chat - scripted AT commands issued to modem for Verizon
e20-leds, e20-buttons packages – a simple LED and button control scripts package	led-1, led-2, led-3	Controls lighting for led 1, 2, and 3
	button-1, button-2, button-3	Reads button states
e20-gpio-scripts package - Initializes GPIO lines (/etc/rc2.d)	S30gpios	Start up script to initialize GPIO lines package

E20 support package for the SM220 must be installed via `apt-get:`

e20-snap-utils package – maintenance and support scripts for SM220 *This package depends on SNAP Connect (See Section 5 – SNAP Connect for installation instructions)*	wake_snap_node	wakes the SM220 (if it was sleeping)
	reset_snap_node	resets the SM220
	flash-bridge	performs maintenance on the SM220

7. User Buttons and LEDs

The E20 has 3 user-programmable buttons, and 3 user-programmable tri-color LEDs.

GPIO 40	LED-1	red
GPIO 41	LED-1	green
GPIO 42	LED-2	red
GPIO 43	LED-2	green
GPIO 44	LED-3	red
GPIO 45	LED-3	green
GPIO 117	Button 1	
GPIO 118	Button 2	
GPIO 119	Button 3	

Each tri-color LED is controlled by 2 GPIO lines (red, green) and the combination of these lines gives 3 colors for each LED (red, green, amber).

For convenience, programs are pre-installed on the E20 to interact with these peripherals:

Controlling LEDs

Provided by package e20-leds:

- `/usr/local/bin/led-1`
- `/usr/local/bin/led-2`
- `/usr/local/bin/led-3`

Usage:

- `led-1 red`
- `led-1 green`
- `led-1 amber`
- `led-1 off`

By default, on power-up all three LEDs will turn on in amber mode, then turn off after Linux has booted.

Reading the buttons (Provided by package e20-buttons):

- `/usr/local/bin/button-1`
- `/usr/local/bin/button-2`
- `/usr/local/bin/button-3`

Prints the button status (1 = up, 0 = pressed,) and returns this number as the exit code

8. Controlling the Internal SNAP Engine

The E20 has the ability to wake and reset the SM220 using two dedicated GPIO pins:

- **Wake Snap Node:** GPIO 33 (tied to pin GPIO_F1 on the SM220)
- **Reset Pin:** GPIO 34

Additionally, maintenance scripts on the E20 control the SM220 beyond interacting with the SNAPpy script over the UART. These scripts allow you to perform recovery maintenance such as erasing an unresponsive SNAPpy image, factory defaulting the NV params, and uploading new firmware.

SNAP maintenance scripts and helper utilities can be installed via:

```
$ sudo apt-get install e20-snap-utils
```

Waking the SM220

First, GPIO_F1 must be set as a wakeup pin on the SM220. To do this, use the wakeupOn function:

```
from synapse.pinWakeup import *
from synapse.platforms import *
# other code
# before putting node to sleep
wakeupOn(GPIO_F1, True, False)
```

When you want to wake the node, call the provided `wake-snap-node` script, which will perform this function for you automatically.

- This script is located at `/usr/local/bin/wake-snap-node`

Resetting the SM220

There is no special setup required to reset the SM220, just call `reset-snap-node` which will briefly drive the reset GPIO low, forcing the SM220 to reset.

This script is located at `/usr/local/bin/reset-snap-node`

Restoring the functionality of an unresponsive SM220

The E20 comes with a recovery utility which is capable of erasing the loaded SNAPpy image, factory defaulting the NV params, and uploading a firmware image.

This script is included in the e20-snap-utils package (see above for installation instructions) and is located at `/usr/local/bin/flash-bridge`

Usage:

- `flash-bridge -e` (to erase the running SNAPpy image)
- `flash-bridge -i imageName` (to upload firmware image “imageName” to the SM220)
- `flash-bridge -nv` (to restore the default NV params)

9. SM220 SNAP Node

The SM220 SNAP node connects the wireless SNAP network to the E20 gateway. The SM220 communicates with the rest of the E20 via 2 serial ports.

LED Control

The SM220 controls the tri-color led labeled “SNAP” on the case via GPIO_A4 (green) and GPIO_A5 (red). (For amber, both GPIOs are used.)

NOTE: This LED is only accessible via the SNAP node.

Example of flashing the led via GPIO_A4 on the SM220:

```
from synapse.platforms import *
@setHook(HOOK_STARTUP)
def startupEvent():
    setPinDir(GPIO_A4, True)
@setHook(HOOK_1S)
def tick():
    pulsePin(GPIO_A4, 500, False)
```

Controlling the E20 from the SM220

The SM220 has two pins dedicated to interfacing with the E20.

- GPIO_C4 – System Reset. Active low. Triggers a hardware reset on the E20.
- GPIO_F2 – Interrupt / General Use – Tied to GPIO 32 on the E20.

GPIO_F2 can be used to signal the E20 that some event of interest has occurred that might need a response even if a SNAP Connect application is not running.

A simple example of how to monitor for this interrupt condition or event is provided below:

```
if [[ ! -d /sys/class/gpio/gpio32/ ]] ; then
    echo 32 > /sys/class/gpio/export
    echo in > /sys/class/gpio/gpio32/direction
fi

while [[ `cat /sys/class/gpio/gpio32/value` != 0 ]] ; do
    sleep 1
done

echo "Got interrupt!"
```

GPIO_C4 can be used to perform a hard reset of the E20 without resetting the SM220. This can be used to force a reboot if you are otherwise unable to access your E20. Note that any saved data may be lost when performing a hard reset, and this is intended only to recover control when the E20 has become unresponsive.

```
@setHook(HOOK_STARTUP)
    setPinPullup(GPIO_C4, True)
    writePin(GPIO_C4, True)
    setPinDir(GPIO_C4, True)
def resetE20():
    pulsePin(GPIO_C4, 1, False)
```

10. Factory Restore / Re-Flashing your E20

For most E20s, a factory restore is accomplished using a microSD card image.

Note: Not all E20s can be programmed via the microSD card. If your E20 case does not look like the one pictured in Figure 1, or when you remove the back panel there is no microSD card slot and button present, skip ahead to **Restoring from a USB Flash Drive**.

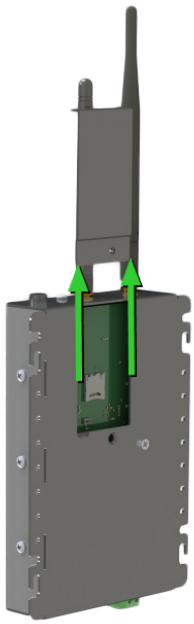


Figure 1: The microSD card slot can be found by removing the back panel.

Restoring from a MicroSD Card

- From the Synapse Wireless forums, download the newest microSD card E20 installer image: https://forums.synapse-wireless.com/showthread.php?t=9
- Write this image to your microSD card.
    - In Windows, this is easiest done using something like Win32DiskImager.
    - In Linux, you can use dd from the command line:

```
$ dd if=e20-VERSION-sdcard.img of=/dev/sdX bs=1M
# where e20-VERSION-sdcard.img is the sdcard image file
# and /dev/sdX is the device file of the uSD
card device
$ sync
$ eject /dev/sdX
```

- Remove the back panel on the E20
- Insert the microSD card into the microSD card slot on the back of the E20
- Hold the button near the slot, apply power, and release the button
- After a few moments, LEDs 1, 2, and 3 should turn red. When programming blink green three times, programming has finished, and you can re-apply power to your unit.

Restoring from a USB Flash Drive

Some very early models of the E20 did not support flashing an image from the microSD card. These devices require a USB flash drive image and slightly more user interaction to flash a new image, however the process is finished, theyvery similar.

- From the Synapse Wireless forums, download the newest USB flash drive E20 installer image: https://forums.synapse-wireless.com/showthread.php?t=9
- Write this image to your flash drive.
    - In Windows, this is easiest done using something like Win32DiskImager.
    - In Linux, you can use dd from the command line:

```
$ dd if=e20-VERSION-sdcard.img of=/dev/sdX bs=1M
# where e20-VERSION-sdcard.img is the sdcard image file
# and /dev/sdX is the device file of the drive - NOT a partition on
the drive!
$ sync
$ eject /dev/sdX
```

- Apply power to the E20
- Establish a serial connection to the E20, then interrupt U-Boot by pressing a key when the prompt “Hit any key to stop autoboot” appears.
    - If you miss this prompt and the E20 boots into Linux, issuing “reboot” from Linux will return you to it.

- Once at the U-Boot-E20> prompt, insert the flash drive into E20 and issue the following two commands:

```
usb start; mw 0x12000000 0x0 0x10000; mmc write 0x12000000 0 0x10000;
fatload usb 0 0x12000000 u-boot.imx; mmc write 0x12000000 2 0x260;

usb start; fatload usb 0 0x12000000 installer-uImage; fatload usb 0
0x18000000 imx6s-e20.dtb; setenv bootargs console=ttymxc0,115200 --no-log;
bootm 0x12000000 - 0x18000000
```

- After a few moments, the LEDs should turn red – when the LEDs blink green three times and you can then re-apply power to your unit.

11. Common Linux Operations

The E20 is pre-loaded with Ubuntu Linux, and use assumes knowledge of the Linux OS. Ample documentation is available online to perform all operations within the capability of the E20. The following are a few common operations:

Making Custom Software run Automatically at Startup

For trivial scripts that should do something once at boot time, you can add a simple shell script you want called at one of a few places:

- `/etc/rc.local`
- `/etc/rc2.d/newprogramname`

For applications you want to have started as a service which can be started, stopped, and restarted; you can create an Upstart service at:

- `/etc/init/newprogramname.conf`

A basic example of making a SnapConnect application run as a service could be achieved by generating the following file in `/etc/init/snapconnect.conf` :

```
# SNAP Connect - start a SNAP Connect application as a service
#

description     "Start SNAP Connect"

start on runlevel [2]
stop on runlevel [!2]

exec python /home/snap/my_snapconnect_example.py
```

The command `python /home/snap/my_snapconnect_example.py` would then be executed on boot, and stopped on shutdown. You could also start and stop by issuing the following commands:

```
$ sudo service snapconnect start
$ sudo service snapconnect restart
$ sudo service snapconnect stop
```

Note that this is a basic example that illustrates a starting point for developing your custom Upstart service.

Resetting a Lost User Password

- Interrupt u-boot by rebooting the E20, then pressing any key when the message “Hit any key to stop autoboot” appears.
- From the U-Boot command prompt, enter the following two commands:

```
U-Boot-E20> setenv mmcargs 'setenv bootargs console=${console},${baudrate} --no-log fec.macaddr=${macaddr} root=${mmcroot} rootdelay=2 rw single'
U-Boot-E20> boot
```

Ensuring that the E20 stays connected over Ethernet

An unaddressed bug in Ubuntu 14.04 can affect E20 units with eth0 configured to boot automatically in DHCP mode. By default, the E20 is patched to activate eth0 in a mode which should put the DHCP client in the background, and obtain an IP address when a link becomes available. However, if you had an IP address before, and the lease on your address has not passed, dhclient only tries to renew it ONE time and then quits trying. (As opposed to if you didn’t have one, it keeps searching for a DHCP server.) This can potentially leave the E20 without an IP address. If this is a concern for you, consider setting up the following script to run in the background on boot:

```
#!/bin/bash
while [[ true ]] ; do
   OUTPUT=`ifconfig eth0 | grep 'inet addr:' | wc -w 2>/dev/null`
   if [[ $OUTPUT == "0" ]] ; then
         ifdown eth0 && ifup eth0 &
   fi
   sleep 600
done
```

12. Wifi

Typical steps in working with wifi:

Edit `/etc/network/interfaces`:

```
#auto wlan0
iface wlan0 inet dhcp
    wpa-conf /etc/wpa_supplicant.conf
    wpa-driver wext
```

- wlan0 does not activate by default
  - Uncomment `#auto wlan0` to activate automatically

Connecting to an access point:

- Generate your passphrase key:

```
$ wpa_passphrase 'myssid' 'mypassword'
network={
        ssid="myssid"
        #psk="mypassword"
psk=2f0568b3492812bd56b946dbaf3fd7dd669b9a4602a09aa6462ff057949b025c
}
```

- Put the output of that into `/etc/wpa_supplicant.conf`

```
network={
     ssid="myssid"
psk=2f0568b3492812bd56b946dbaf3fd7dd669b9a4602a09aa6462ff057949b025c
}
```

- You may need additional options depending on your network setup.
- You can now either run `sudo ifup wlan0` to bring up the interface and connect to the network, or reboot. Once `auto wlan0` is present in your network interfaces file, wlan0 will be brought up and attempt to connect to the network automatically on boot.

Setting up Access-Point (AP) mode:

- `apt-get install udhcpd`
- Generate your passphrase (you will need this information later when we call `iwpriv`):

```
$ wpa_passphrase 'myssid' 'mypassword'
network={
   ssid="myssid"
   #psk="mypassword"
   psk=2f0568b3492812bd56b946dbaf3fd7dd669b9a4602a09aa6462ff057949b025c
}
```

- Set up `/etc/udhcpd.conf`

```
# The start and end of the IP lease block
start          192.168.0.20    #default: 192.168.0.20
end            192.168.0.254   #default: 192.168.0.254
# The interface that udhcpd will use
interface      wlan0           #default:eth0
[rest cut off for brevity]
# The start and end of the IP lease block
start          192.168.0.20    #default: 192.168.0.20
end            192.168.0.254   #default: 192.168.0.254
# The interface that udhcpd will use
interface      wlan0           #default:eth0
[rest cut off for brevity]
```

Set up udhcpd to run by default by editing `/etc/default/udhcpd`:

```
# Comment the following line to enable
DHCPD_ENABLED="yes"
# Options to pass to busybox' udhcpd.
#
# -S    Log to syslog
# -f    run in foreground
DHCPD_OPTS="-S"
```

Set up wlan0 with a static IP so it can act as a gateway by editing `/etc/network/interfaces`:

```
# interfaces(5) file used by ifup(8) and ifdown(8)
# Include files from /etc/network/interfaces.d:

allow-hotplug eth0
iface eth0 inet dhcp

auto wlan0
iface wlan0 inet static
    address 192.168.0.1
    netmask 255.255.255.0
```

Set things up with `iwpriv` (using the information you gathered from wpa_passphrase earlier,):

```
snap@localhost:~$ sudo iwpriv wlan0 AP_SET_CFG
ASCII_CMD=AP_CFG,SSID="myssid",SEC="wpa2-
psk",KEY=2f0568b3492812bd56b946dbaf3fd7dd669b9a4602a09aa6462ff057949b025c,CHANNE
L=1,PREAMBLE=1,MAX_SCB=8,END
```

Change parameters as needed (e.g. what CHANNEL you want to be on.)

Start AP mode:

```
snap@localhost:~$ sudo iwpriv wlan0 AP_BSS_START
```

Stop AP mode:

```
snap@localhost:~$ sudo iwpriv wlan0 AP_BSS_STOP
```

13. Cell Modem

The E20 currently supports the Telit DE910-DUAL on Verizon Wireless. Support for AT&T and others is forthcoming.

To initiate a PPPD session on Verizon Wireless:

```
/usr/local/bin/callvz &
```

To terminate the connection:

```
poff telit-verizon
```

These scripts are not guaranteed to work with your network or data plan, and are provided for illustrative purposes only.

Troubleshooting Cellular Connectivity

If you are deploying the E20 in a situation which will be dependent on the cellular connection for connectivity, you will most likely want to take some precautions to ensure that the connection re-establishes itself in the event of failure (signal and handshaking issues, etc.) There are multiple ways to potentially address this, and the best way is largely depended on your needs and setup.

- The addition of a 'persist' string to your PPP configuration file, will make PPPD attempt to reconnect if it detects the connection to the tower has been dropped. The PPPD session initiated by `callvz` uses `/etc/ppp/peers/telit-verizon` as the PPP configuration file.
- A background shell script, monit utility, or cron job which monitors if PPPD is running, and re-launches it if it detects it is not
- Use of the reset-cell-modem script, which pulls the reset line on the cell modem, hardware resetting it if needed.

14. Troubleshooting and Common Problems

The Ethernet does not work or eth0 does not appear in ifconfig

Most likely, your MAC address has not been set. If you run ifconfig -a and see output similar to:

```
eth_badmac      Link encap:Ethernet  HWaddr 00:1c:2c:ff:ff:ff
```

Your MAC address has reverted to the default and needs to be set. See **SNAP Connect is not working** below.

SNAP Connect is not working

Check for one of the following possible issues:

1. SNAP Connect may be out of date – prior versions of SNAP Connect were incompatible with some version of the Python Tornado package or may have failed to obtain the MAC address correctly. Upgrade to the latest version of SNAP Connect by using the instructions in Section 5.
2. Your MAC address is not set:
   - Start or reboot your E20, and press a key when prompted by U-Boot to interrupt autoboot.
   - You should arrive at the `U-Boot-E20>` command prompt.
   - Enter the following commands:

     ```
     U-Boot-E20> setenv macaddr "0x00,0x1c,0x2c,0xXX,0xXX,0xXX"
     U-Boot-E20> saveenv
     U-Boot-E20> saveenv
     ```

   - Replace the 0xXX entries with your last three octets of your MAC address, which should be found on the label on your E20's case. Note that the comma delimited 0xXX format is required.

I cannot SSH into my E20

You cannot SSH into the E20 as root, or any user account which does not have the password set. Be sure to have set a password for the account you want to use to connect.

Ensuring that the E20 stays connected over Ethernet

An unaddressed bug in Ubuntu 14.04 can affect E20 units with eth0 configured to boot automatically in DHCP mode. By default, the E20 is patched to activate eth0 in a mode which will put the DHCP client in the background and obtain an IP address when a link becomes available. However, if you had an IP address before, and the lease on your address has not passed, dhclient only tries to renew it ONCE and quits trying. (As opposed to if you didn't have one, it keeps searching for a DHCP server.) This can potentially leave the E20 without an IP address. If this is a concern for you, consider setting up the following script to run in the background on boot:

```
#!/bin/bash
while [[ true ]] ; do
   OUTPUT=`ifconfig eth0 | grep 'inet addr:' | wc -w 2>/dev/null`
   if [[ $OUTPUT == "0" ]] ; then
         ifdown eth0 && ifup eth0 &
   fi
   sleep 600
done
```

15. Regulatory Information and Certifications

RF Exposure Statement

This equipment complies with FCC radiation exposure limits set forth for an uncontrolled environment. This equipment should be installed and operated with minimum distance of 20cm between the radiator and your body. This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.

FCC Certifications and Regulatory Information (USA Only)

FCC Part 15 Class B

These devices comply with part 15 of the FCC rules. Operation is subject to the following two conditions: (1) These devices may not cause harmful interference, and (2) These devices must accept any interference received, including interference that may cause harmful operation.

Radio Frequency Interference (RFI) (FCC 15.105)

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that the interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Reorient or relocate the receiving antenna. Increase the separation between the equipment and receiver. Connect the equipment into an outlet on a circuit different from that of the receiver. Consult the dealer or an experienced radio/TV technician for help.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Labeling Requirements (FCC 15.19)

This device complies with Part 15 of FCC rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

If the FCC ID for the module inside this product enclosure is not visible when installed inside another device, then the outside of the device into which this product is installed must also display a label referring to the enclosed module FCC ID.

Modifications (FCC 15.21)

Changes or modifications to this equipment not expressly approved by Synapse Wireless, Inc. may void the user's authority to operate this equipment.

Declaration of Conformity

(In accordance with FCC 96-208 and 95-19)

Manufacturer's Name: Synapse Wireless, Inc.

Headquarters: 6723 Odyssey Drive

Huntsville, AL 35806

Synapse Wireless, Inc. declares that the product:

Product Name: E20-0, E20-3G, E20-3G1

to which this declaration relates, meet the requirements specified by the Federal Communications Commission as detailed in the following specifications:

- Part 15, Subpart B, for Class B equipment
- FCC 96-208 as it applies to Class B personal computers and peripherals

The products listed above have been tested at an External Test Laboratory certified per FCC rules and has been found to meet the FCC, Part 15, Emission Limits. Documentation is on file and available from Synapse Wireless, Inc.

Industry Canada (IC) Statement

This Class B digital apparatus complies with Canadian ICES-003.

Cet appareil numérique de la classe B est conforme à la norme NMB-003 du Canada.