ControlTrac
BY DYACON®

** Preliminary **

CT650

ControlTrac Computer (Linux OS)

57-6075 Rev B

This page intentionally left blank.

Contents

NOTICES

© Copyright 2020 Dyacon, Inc.

Manufacturer

Dyacon, Inc.
1770 Research Park Way, Suite 168
North Logan, UT 84341
USA

Declarations

ControlTrac™ CT650 is a low-power electronic industrial device.

RoHS

All electronic and mechanical components conform to RoHS, Directive 2002/95/EC.

FCC CFR Part 15

This equipment complies with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a commercial installation.

Warranty Information

Limited Hardware Warranty

Dyacon, Inc. warrants that all ControlTrac products and components shall be free from defects in materials and workmanship for a period of one (1) year from the date of shipment when installed according to instruction manuals accompanying said hardware and used for the purpose for which said hardware was designed. In the event a defect in materials or workmanship is discovered and reported to Dyacon within the warranty period, Dyacon will at its option repair the defect or replace the defective product. This warranty does not apply where the product has been operated outside the specifications of the product. Dyacon's obligation hereunder will be limited to such repair or replacement. Customers shall have the responsibility to ship the defective equipment to Dyacon at its (customer's) expense, with all cost of shipment prepaid. Dyacon will ship the repaired or replaced item at its (Dyacon's) expense using the preferred shipment method of Dyacon.

Disclaimer of Warranties

The warranties set forth above are in lieu of all other warranties of Dyacon, whether written, oral, or implied. Dyacon makes no warranties regarding its products (hardware or software), including without limitation warranties as to merchantability, fitness for a particular purpose, any warranty arising from course of performance, course of dealing or usage of trade whether any of the foregoing warranties are either expressed or implied. Dyacon specifically makes no warranties as to the suitability of its products for any particular application. Dyacon shall in no event be liable for performance, or use of any product covered by this agreement whether such claim is based upon warranty contract (express or implied), strict liability, negligence, or otherwise. Any responsibility and/or liability of Dyacon shall, in connection with a warranted product, be limited in maximum amount to the original purchase price of that product.

Removal of Serial Number

Removal of the original serial number label or reprogramming of the electronic serial number voids any warranty on the device. Dyacon will not repair or update products if the serial number label missing or legitimate ownership cannot be verified. Dyacon may not return equipment that is missing serial numbers or where legitimate ownership is in question.

Updates or Modifications

Dyacon shall be under no obligation to update or modify its products except as herein noted to correct defects or errors. Customer agrees that all representation and warranties contained herein

shall be immediately null and void in the event of incorrect installation, modification, alteration, or change in or to any product affected by or on behalf of customer except for a change made by Dyacon or other direct supervision thereof.

CT650 Introduction

Scope

The content of this document is intended for developers, integrators, and installers of CT650.

This document does not include information related to integration into vehicle systems. Integrators and installers are responsible for observing proper electrical and electronic standards and applicable safety measures when connecting CT650 to vehicle systems.

Technical Support

Contact Information

Dyacon, Inc.

1770 Research Park Way, Suite 168
North Logan, UT 84341

Phone: (435) 753-1002

Email: support@dyacon.com

Internet: www.dyacon.com

Normal business hours are from 8:00 am to 5:00 pm. (Mountain Time Zone, GMT -0700)

Phone / Email Support

If you need technical support via the phone or email, please have the following information ready:

Product name, model number, and serial number.

Your name and name of the purchaser of the equipment.

Name of company, institution, or agency.

Phone number, email address.

Billing and Shipping address.

A clear description of the question or problem.

Product Overview

Features

CT650 uses an adaptable board configuration that allows the device to be customized to meet the price and feature needs of a wide range of integrators. This manual covers all of the available hardware options, some of which may not be available on custom board options.

Standard Features

Linux operating system

Console port for development

Digital input (3x)

Digital output (3x)

Ethernet

Can-bus

RS-485

RS-232

Optional Features

Audio in and out

Push-to-talk (Digital output 4) transmitter activation

Uninterruptible power supply

Cell phone

WiFi

GPS

Real-time clock with backup battery

Micro SD card holder

Micro USB

Electrical Architecture

CT650 uses a single circuit board for all processor and I/O options. The wireless module (cell phone or WiFi) is a plug-in option. This allows for technology upgrades and easy service.

Vehicle power is supplied directly to the main circuit board.

Mechanical Architecture

A splash-resistant, compact enclosure includes rugged mounting features and sealed connectors.

SMA connectors are used for both cell phone and GPS.

Connectors

External Connectors A (Grey) and B (Black)

The external connectors are used for attaching CT650 to external power and signals in a vehicle. The connectors are automotive-style, providing sealing and vibration resistance for each wire.

Mating Connector Components

The external connectors are Deutsch DTM series and can be found at https://laddinc.com and other distributors.

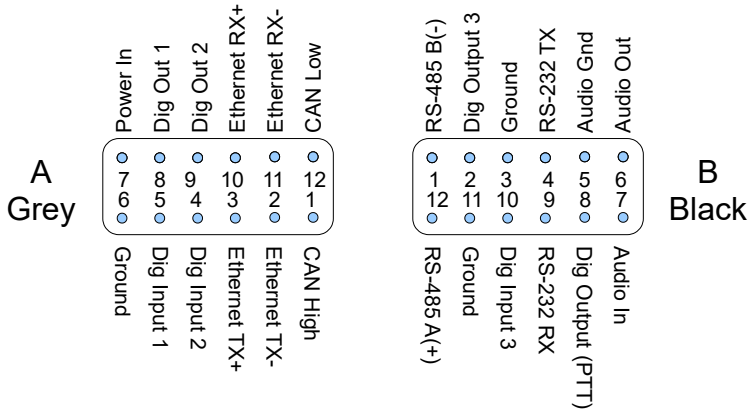
Connector A, Grey: Deutsch DTM06-12SA

Connector B, Black: Deutsch DTM06-12SB

Contacts (receptacles): 1060-20-0122 (Stocked by Dyacon. 22-16 AWG wire)

Alternates: 0462-201-20141, 0462-201-2031, 1062-20-0144, 1062-20-0222

Pin Out



Connector:Pin	Signal	Description
A:1	CAN High	
A:2	Ethernet TX-	
A:3	Ethernet TX+	
A:4	Digital Output 2	
A:5	Digital Output 1	
A:6	Ground	
A:7	Power Input	

A:8	Digital Output 1	
A:9	Digital Output 2	
A:10	Ethernet RX+	
A:11	Ethernet RX-	
A:12	CAN Low	

Connector:Pin	Signal	Description
B:1	RS-485 B-	
B:2	Digital Output 3	
B:3	Ground	
B:4	RS-232 TX	
B:5	Audio Ground	
B:6	Audio Out	
B:7	Audio In	
B:8	Digital Output 4 (PTT)	
B:9	RS-232 RX	
B:10	Digital Input 3	
B:11	Ground	
B:12	RS-485 A+	

Console Port

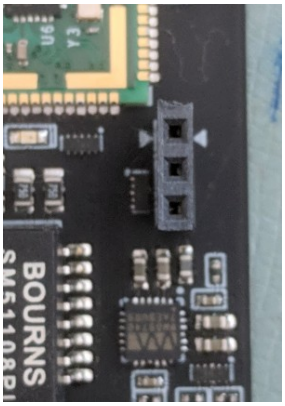
Programming and developer interaction with the Linux operating system is done through the console port. The console port is located on header by the main connector.

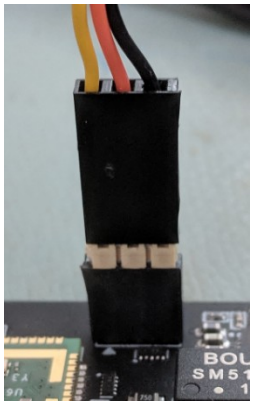
Pin 1 is indicated by a small white triangle.

Signals levels are 3.3 volt UART TTL. A signal conver ter is needed in order to connect the console port to a PC.

Dyacon has used FTDI TTL-232R-3V3 USB to UART converter with a modified connector. These are available from electronic component distributors, such as Mouser and Digikey. The converter is supplied with a six-pin receptacle. This must be modified to the following.

Pin Number	Processor Pin
1	TX (Output)
2	RX (Input)
3	GND





Contact Dyacon if you would like us to supply you with a modified converter.

Serial Port Settings

The default port settings for the console port are:

Baud rate: 115200

Data Bits: 8

Parity: None

Stop Bits: 1

Flow Control: None

Developer Information

Development Environment

Kernel version

Linux 4.14.73
U-boot 2018.07

Development Tools

The development tools have been build using Ubuntu 18.04.

The development toolchain can be obtained at
https://drive.google.com/open?id=1K9EeAicM1Nnls0qH-MTjmZF0cGxpjkSt

Extract the toolchain to your desired location. After doing so, run the relocate-sdk.sh script to update the paths to the new location.

The simplest way to use the toolchain is to add the {extraction directory}/bin/ to your PATH environment variable and then to use arm-linux-gcc, arm-linux-ld, etc.

Example:

```
mkdir ~/CT650-toolchain
tar -xvf CT650-toolchain.tar.xz -C ~/CT650-toolchain/
cd ~/CT650-toolchain
./relocate-sdk.sh
cd ~/testapp/
export PATH="/home/{user}/CT650-toolchain/bin:$PATH"
arm-linux-gcc hello.c -o hello
```

Console Port

Login Information

User: root

Password: OneRainCt650

File System

The system contains the following partitions:

mmcblk0p2 – boot partition A
mmcblk0p3 – boot partition B
mmcblk0p5 – key storage – Mounted at /key
mmcblk0p6 – rootfs partition A – Mounted at /
mmcblk0p7 – rootfs partition B
mmcblk0p8 – app partition A – Overlayed at /app, /etc, and /lib in rootfs partition A
mmcblk0p9 – app partition B – Overlayed at /app, /etc, and /lib in rootfs partition B
mmcblk0p10 – data – Mounted at /data

When received the system is configured to use boot partition 1 and rootfs partition 1. Application executable should be placed in the rootfs partition. Application data should be placed in the data partition which is mounted at /data.

The system has dual boot and rootfs partitions to allow the swupdate tool to do a dual image update.

Dual Boot Partitions

NOTE: After boot, the boot count variable located at `0xF804540C` needs to be cleared by writing a value of `0xB0010000`. This can be done in the user's application or from the command line using:

```
devmem 0xF804540C 32 0xB0010000
```

Building from source

Buildroot is used to build the OS and root file system for the CT650. As the CT650 is sold only to integrators, the modifications done for the CT650 is not public. Please contact your integrator to get a copy of the open source files used for your certain version of CT650. If you are an integrator please contact Dyacon to receive an invitation to the github repo.

To configure Buildroot for the CT650, use the following instructions.

```
cd ~/
git clone https://github.com/ericrjohn/ct650.git
git clone https://github.com/linux4sam/buildroot-external-
microchip.git
cd buildroot-external-microchip/
```

```
git checkout linux4sam_6.0 -b buildroot-external-microchip-
linux4sam_6.0
cd ../ct650/
git checkout buildroot-at91-linux4sam_6.0
BR2_EXTERNAL=../buildroot-external-microchip/ make
atmel_sama5d27_som1_ct650_defconfig
```

Build the system.

```
make
```

Configuration changes can be made using menuconfig.

```
make menuconfig
```

Sometimes the changes made will require a cleaning to build properly. If make fails after a change, clean and try again. When removing a package, it is required to do a clean to remove the output files from the root filesystem.

```
make clean
```

More information

https://www.at91.com/linux4sam/bin/view/Linux4SAM/BuildRootBuild#How_to_build_Buildroot_for_AT91

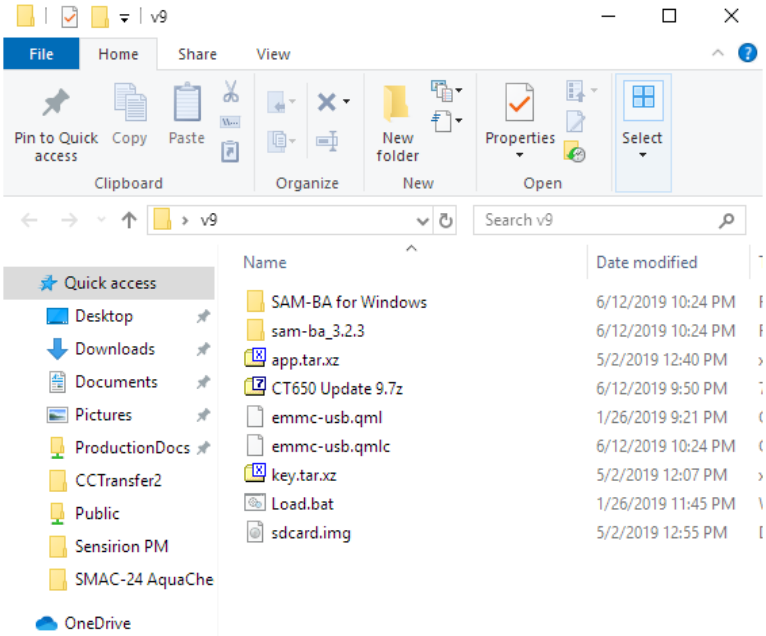
Updating the OS on the CT650

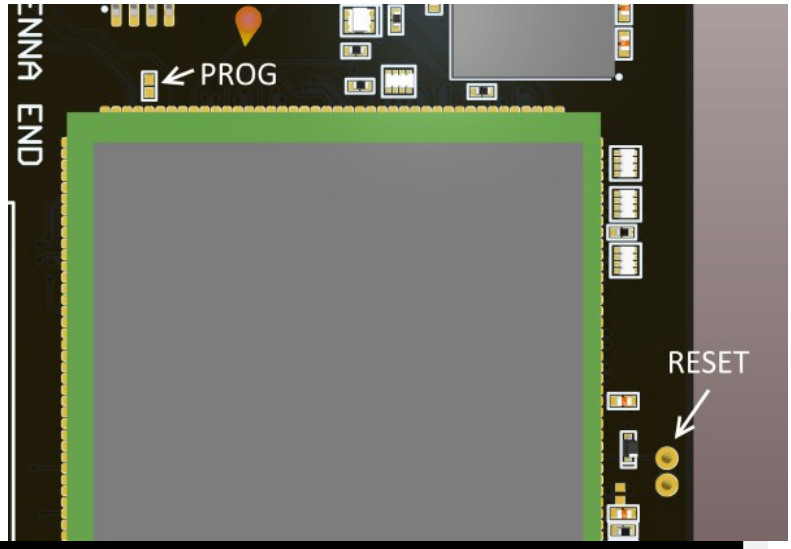
Obtain the latest image from https://drive.google.com/open?id=10i_kAFYX38-EmtCGV219I2o1ajRuZjvq

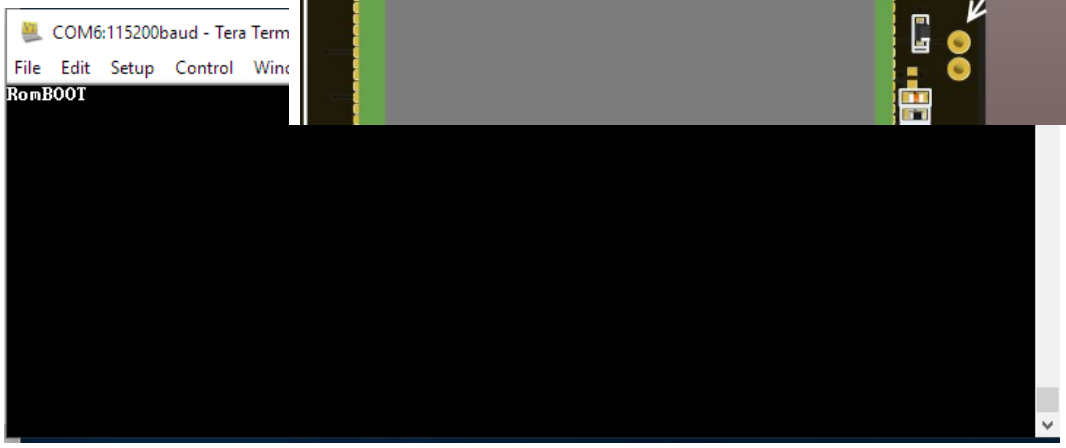
Update the CT650 using the following process:

1. Download and extract the latest image.
2. Connect to the console port using a terminal program as described in the Console Port section of this document.
3. Attach a USB micro cable to the unit using the connector on the back edge of the board. If your computer needs drivers, the drivers are located in sam-ba_3.2.3\driver.
4. Remove the circuit board assembly from the case.
5. Using tweezers ora similar tools, activate programming mode by shorting the PROG resistor pads while shorting and releasing the RESET pads.

   Ensure that the short is held on the PROG pins until 'RomBOOT' is seen on the terminal and booting stops.

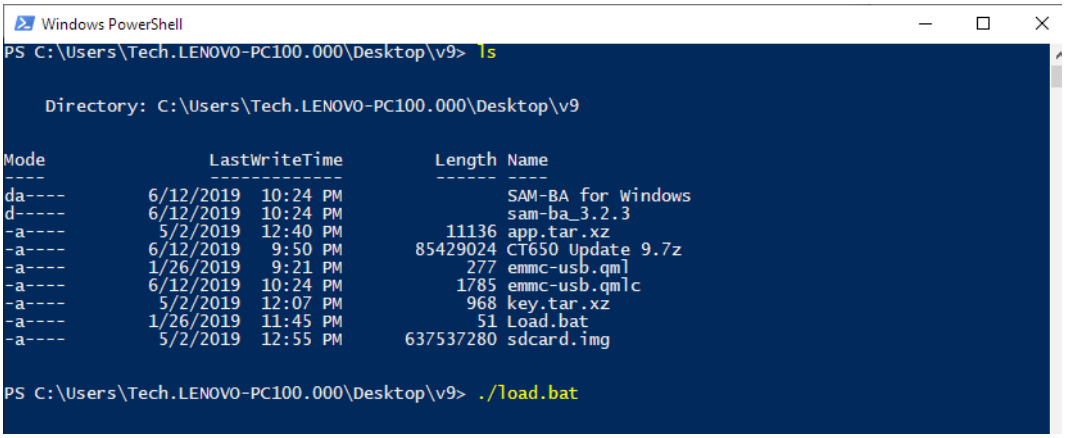


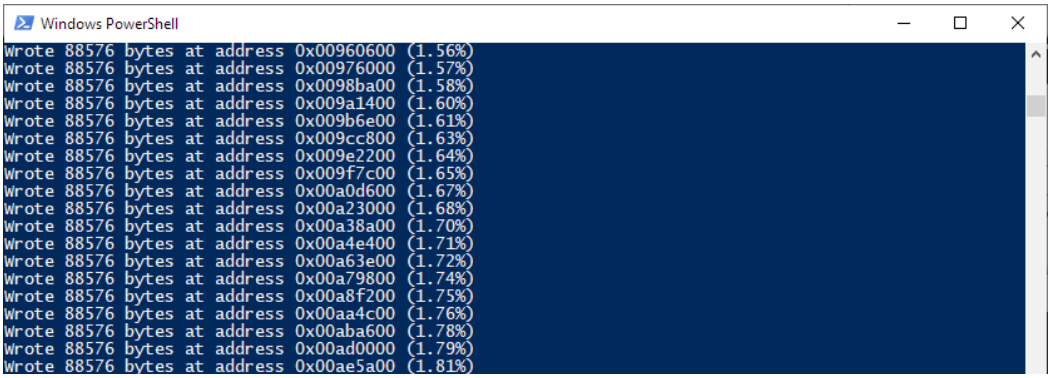




6. Using PowerShell (Shift+Rt Click in directory), run the following commands.

```
Run "0 - Load qspi.bat"

Run "1 - Set Boot qspi.bat"

Run "2 - Load emmc.bat"

Boot unit

Format data partition:
    cd /
    mkfs.ext4 /dev/mmcblk0p10
    mount data
    cd /data
    chmod 755 .
```





Allow the loading to continue until you see:

```
Press any key to continue . . .
```

7. Reset or reboot CT650.
8. Depending on the update, the data partition will need to be formatted and the unit rebooted before it will mount.

```
mkfs.ext4 /dev/mmcblk0p10
```

LED Control

Turn on:

```
echo 1 > /sys/class/leds/red/brightness
echo 1 > /sys/class/leds/green/brightness
```

Turn off:

```
echo 0 > /sys/class/leds/red/brightness
echo 0 > /sys/class/leds/green/brightness
```

Digital Outputs

Digital outputs are controlled from the following microprocessor pins:

Output Number	Processor Pin	GPIO Number
1	PC18	82
2	PC25	89
3	PC16	80
4	PC11	75

sysfs option

Setup output 1:

```
echo 82 > /sys/class/gpio/export
echo out > /sys/class/gpio/PC18/direction
```

Turn on:

```
echo 1 > /sys/class/gpio/PC18/value
```

Turn off:

```
echo 0 > /sys/class/gpio/PC18/value
```

libgpiod option

The libgpiod option is the way Linux is moving towards and sysfs is being depreciated. More info can be found here:
https://linuxpiter.com/system/attachments/files/000/001/532/original/Linux_Piter_2018_-_New_GPIO_interface_for_linux_userspace.pdf

```
gpioset gpiochip0 82=0 89=0 80=0 75=0
```

NOTE: While command line usage is shown, when executing from an application libgpiod C API can be used.

Digital Inputs

Digital Inputs and their pull resistors are controlled from the following microprocessor pins:

Input Number	Processor Pin	GPIO Number
1	PC12	76
2	PC17	81
3	PC21	85

Input Pull Number	Processor Pin	GPIO Number	Default
1	PC9	73	Pulled High
2	PC13	77	Pulled High
3	PC19	83	Pulled High

sysfs option

Setup input 1:

```
echo 76 > /sys/class/gpio/export
echo 1 > /sys/class/gpio/PC12/active_low
```

Read:

```
cat /sys/class/gpio/PC12/value
```

Setup input pull 1:

```
echo 73 > /sys/class/gpio/export
echo out > /sys/class/gpio/PC9/direction
echo 1 > /sys/class/gpio/PC9/active_low
```

Pull high:

```
echo 1 > /sys/class/gpio/PC9/value
```

Pull low:

```
echo 0 > /sys/class/gpio/PC9/value
```

libgpiod option

```
gpioset -l gpiochip0 73=1 77=1 83=1
gpioget -l gpiochip0 76 81 85
```

Power-Good Signal

The Power-Good signal is used to indicate power stability. When the power-Good signal is low, the application should complete any flash memory activities and sleep or shut down.

Signal	Processor Pin	GPIO Number
Power Good	PB21	53
Capacitor Good	PB24	55

```
gpioget gpiochip0 53 55
```

Serial Ports

The following serial ports are available on the unit:

Port	Device
RS485 Port 1	/dev/ttyS1
RS485 Port 2	/dev/ttyS3
RS232	/dev/ttyS2
Cell UART Port	/dev/ttyS4

CAN Bus Port

Bring port up

```
gpioset gpiochip0 122=0 (Enable driving the bus - This should be
automatic in the future)
ip link set can0 up type can bitrate 1000000
```

Testing

NOTE: It is critical to have a termination resistor on the bus while operating.

On DUT

```
cangen -v can0
```

On Host (unit used as a test fixture)

```
candump can0
```

Audio

Audio support will be through ALSA.

Currently the codec is not functional. More info to come.

Sleep

The following Linux command can be used to initiate sleep. The real-time clock (RTC) and digital inputs can be used as a wake source.

```
echo mem > /sys/power/state
```

The following instructions use *rtcwake*, which is on CT650.
https://www.linux4sam.org/bin/view/Linux4SAM/UsingUltraLowPowerMode1#RTC_alarm_to_wake_from_the_sleep

Due to multiple RTC's something like this might be needed.

```
# echo enabled > /sys/class/rtc/rtc0/device/power/wakeup
# rtcwake -m mem -d rtc0 -s 10
```

Developers might find section 3.2 of the following manual might helpful.
http://ww1.microchip.com/downloads/en/Appnotes/AN_3251-How-to-Use-SAMA5D2-RTC-Under-Linux-00003251a.pdf

Wake from the digital inputs is possible. If this wake mode is needed, contact Dyacon to update the device tree to enable them as wake up sources.

Cell Phone Module

The CT650 supports Telit cellphone modules manufactured by NimbeLink. While efforts have been made to include support for all the different Telit modules, there might be some option not supported. Please let us know if something is missing.

Please see the Telit Modules Linux USB Drivers User Guide for more information.

https://www.telit.com/wp-content/uploads/2017/10/1VV0301371_Telit_Modules_Linux_USB_Drivers_User_Guide_r3.pdf

NOTE: The following was tested using a NL_SW_LTE_TSVG_B module.

GPIO Pins used for module control

Function	Processor Pin	GPIO Number
Cell Reset	PB2	34
Cell Power Control	PB8	40
Cell USB Disconnect	PB6	38

Powering module

The module can be powered by a pulse on the power control pin.

```
gpioset -m time -s 1 gpiochip0 40=1
gpioset gpiochip0 40=0
```

After powering up the module, enable the USB connection.

```
gpioset gpiochip0 38=1
```

After a few seconds there should be several USB devices registered (depending on module).

Powering module down

It is recommended to power module down before cutting power. This can help the module not to become corrupted. This can also be done using an AT command to the module.

```
gpioset -m time -s 3 gpiochip0 38=0 40=1
gpioset gpiochip0 40=0
```

The module can also be shut down by sending the shutdown command at#shdn to the module. After five to ten seconds the module should be powered down.

Testing UART connection

```
microcom -s 115200 /dev/ttyS4
at
(OK returned)
(use ctrl-x to exit)
```

Activating network connection using qmicli

If the module is configured to support a network adapter, a new network interface named wwan0 should be present.

```
ifconfig -a
(should see wwan0 interface)
```

Depending on the module, you can use the following commands to start the network connection. The following is for the NimbeLink NL-SW-LTE-TSVG module.

```
ifconfig wwan0 up
qmicli -d /dev/cdc-wdm0 --wds-start-network="ip-type=4" \
    --client-no-release-cid
udhcpc -i wwan0
```

If the module in use does not come preprogramed with the correct APN settings, add the APN setting to the qmicli command as shown:

```
qmicli -d /dev/cdc-wdm0 --wds-start-network="ip-
type=4,apn=vzwinternet" \
    --client-no-release-cid
```

At this point the data connection through the cell phone should be functional.

Cell Phone Module Information

https://nimbelink.com/Documentation/Development_Kits/NL-SWDK/30052_NL-SWDK_SkywireQMIAppNote.pdf

Activating network connection using PPP

Linux pppd can also be used for the network configuration. The following is for the NimbeLink NL-SW-LTE-TSVG module. Other modules can be supported by changing the script files. See the more information section below.

Create a file located at /etc/ppp/peers/vzw-TSVG.

```
mkdir -p /etc/ppp/peers
vi /etc/ppp/peers/vzw-TSVG
```

And place the following in the file:

```
/dev/ttyUSB3
115200
connect "/usr/sbin/chat -v -f /etc/ppp/peers/vzw-TSVG-chat"
noauth
defaultroute
usepeerdns
local
debug
updetach
```

Create another file located at /etc/ppp/peers/vzw-TSVG-chat.

```
vi /etc/ppp/peers/vzw-TSVG-chat
```

And place the following in the file:

```
TIMEOUT 35
ECHO ON
'' \rATZ
OK 'ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0'
OK AT+CGDCONT=3,"IPV4V6","VZWINTERNET","",0,0
```

```
OK ATD*99***3#
CONNECT ''
```

Start the connection with the following command.

```
pon vzw-TSVG
```

At this point the data connection through the cell phone should be functional.

Cell Phone Sleep

The cell phone has several power modes. Depending on the functions that your applications requires during system sleep, the cell phone will to be set accordingly.

AT+CFUN=5

Releasing the cell phone DTR line will allow it to enter a lower power sleep mode.

*** More information needed. ***

More Cell Phone Module Information

https://nimbelink.com/Documentation/Skywire/4G_LTE_Cat_3/30058_NL-SW-LTE-TSVG_PPP.pdf

*** Link to AT Command manual needed. ***

GPS

Some cell phones contain GPS units. This can be enabled from one of the unused serial ports (/dev/ttyS4, /dev/ttyUSB2, /dev/ttyUSB3 for the NL-SW-LTE-TSVG).

More information

https://nimbelink.com/Documentation/Skywire/4G_LTE_Cat_3/30104_NL-SW-LTE_GPSAssistedGPSAppnote.pdf

Application

The CT650 is configured to start an application contained in the /app directory named mpu-client.

If the application exits (either normally or from a crash), the CT650 will restart the application.

Software Updating

There are multiple methods of updating the software on the CT650. Which one depends on how involved the update is and what needs updating.

Simple application updating

Before the CT650 starts the application, it checks for an update file located at /update/updateapp.tar.xz. If the file is present, it will extract the archive into the /app directory before starting the application.

After extracting the archive, if a script called PostProcess is found in the /app directory it will run this script before completing the update. After running the script it will delete the PostProcess script so it only runs once.

Once the update is complete it will rename the updateapp.tar.xz archive to updateapp.old.tar.xz

This method requires the application to detect, get, decrypt (if desired), check signatures (if desired), and place the final update file in the /update directory. The application then exits itself, the update happens and the new application starts.

While it is possible to update the Linux kernel and a replace the complete root file system using this method, those major updates are better handled by swupdate.

Swupdate updating

Using swupdate, anything on the system can be updated from a single file to the complete system image.

https://docs.updatefactory.io/devices/linux/update-files/

Issues

None

Revision History

Rev	Description	Author	Date
A	Initial Release	E. Bodrero	14APR2014
B	Add Sleep section.		