

MBARI DIC Controller Hardware Checkout Guide

Contents

- [Materials](#)
- [Board Preparation](#)
- [Firmware Installation](#)
- [Test Procedures](#)
- [Pre-test](#)
- [Console and Host Port](#)
- [Licor Serial Port](#)
- [Set Real Time Clock](#)
- [Analog Inputs](#)
- [Battery Voltage Input](#)
- [Valve Ports](#)
- [Pump Ports](#)
- [SD Card](#)

Materials

- PCB
- fuses (BK/PCS 1-5A slow blow)
- RTC backup battery (CR1620 or equivalent)
- SD card (16GB micro SDHC UHS-1 or better)
- console cable
- host cable
- DMM or oscilloscope
- 1x 3-pin ADC/SV test cable
- 1x 2-pin Pump test cable
- 10-20k pot
- 2x MM jumper wire(s)
- 2x MF jumper wire(s)
- test clips
- F-F DB9 serial loop back connector or Phoenix w/ comms looped back
- 12V 2A power supply
- Microchip ICD4 programmer w/ RJ45 programming cable
- MPLAB host computer (OSX/Win/Linux)
- MPLABX IDE v5.40
- XC16 v1.50
- Code Configurator Plugin 3.95.0
- IDE Platform Plugin 1.29
- device pack PIC24F-GA-GB_DFP 1.1.74
- PIC24/dsPIC33/PIC32MM MCU peripheral library 1.167.0
- 16-bit bootloader v1.17.2

Optional * Java (for Unified Host app if installing serial bootloader)

Resources

- [controller PCB schematic](#)
- [test cables](#)
- [command reference](#)

Board Preparation

- install the RTC backup battery

BT1 on left end of PCB
'+' side up
The solder pads lift easily, don't lift retainer clip.

- install the fuses

sockets F1, F2 on PCB
upper right, near console and host cables

- connect cables
 - the ICD4 programming cable to the PCB
 - connect the ICD4 USB cable to the MPLAB host computer
 - connect the console cable power leads to the power supply
- get the firmware repo (either option)
 - clone firmware git repo

```
git clone git@bitbucket.org:mbari/carbonwg-pic.git
```

- checkout the binlog branch

```
$ cd carbonwg-pic  
$ git checkout feature/binlog
```

- install system files on the SD card
 - use an SD Card reader to mount the SD card
 - copy the system files from carbonwg-pic/sdc to the SD card

```
SYSBOOT.SCR  
CONFIG.TXT  
HELP.MD  
DIC2.SCR
```

- edit CONFIG.TXT (on the SD card) and set the serial number field
- if using OSX or linux, run unix2dos on the files to make sure have correct line endings

```
$ dos2unix path/to/sdcard/*
```

- install the SD Card in the PCB socket

Firmware Installation

There are two firmware options

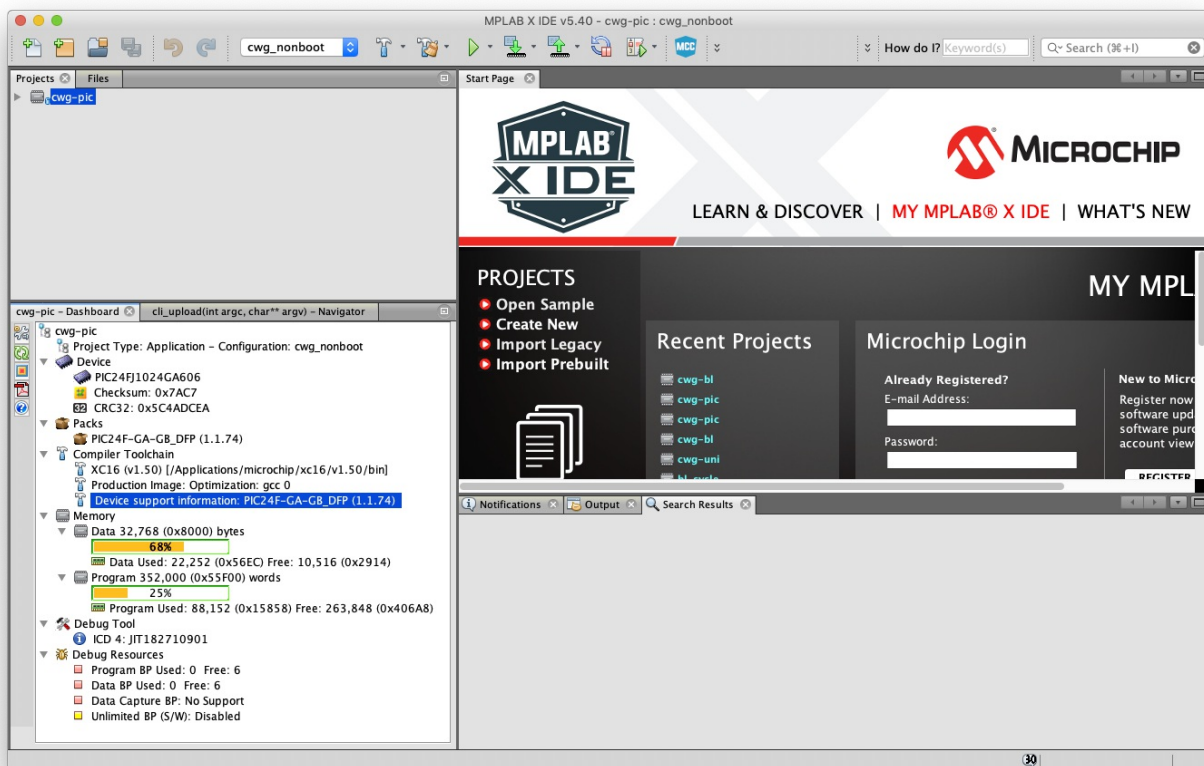
- stand-alone
 - simpler to install
 - firmware updates require ICSP programmer, e.g. ICD4
- with serial bootloader
 - firmware updates may be done via console port
 - requires Microchip Unified Host Java app (distributed with firmware repo)

procedure: Install stand-alone

- open the pic firmware project

File > Open Project... > carbonwg-pic/cwg-pic.X

- select the cwg_nonboot configuration



[carbonwg project non-boot configuration]

- clean/build
- make and program device

procedure: Install serial bootloader version

Installing the serial bootloader version is a three step process:

- build and install the bootloader firmware
- build the carbonwg firmware (uses a different configuration than stand-alone)
- use the bootloader together with the Microchip Unified Host app to load the carbonwg firmware

Step 1: build and install the bootloader firmware

- open the bootloader firmware project

File > Open Project... > carbonwg-pic/cwg-bLX

- clean/build
- make and program device

Step 2: build and install the carbonwg firmware (bootloadable configuration)

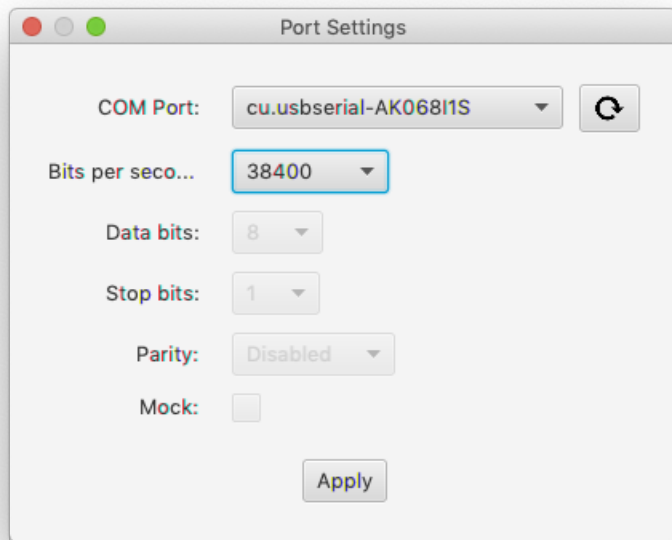
- open the bootloader firmware project

File > Open Project... > carbonwg-pic/cwg-pic.X

- select the cwg_boot configuration
- clean/build
- **WAIT** (do not make and program device)

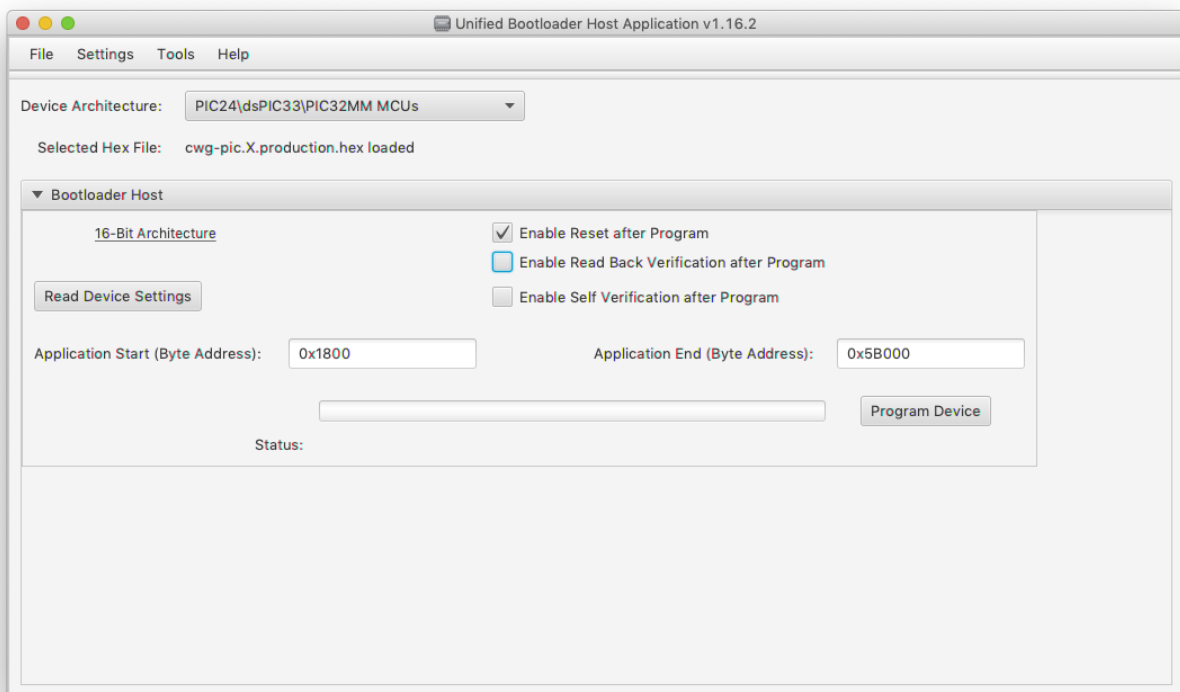
Step 3: Use the bootloader and Unified Host app to load the carbonwg firmware

- unzip the Unified host file carbonwg-pic/utis/unifiedhost- -bin.zip creates directory UnifiedHost-
- double click the jar file to run the app UnifiedHost- /UnifiedHost-.jar
- configure the Device Architecture PIC24/dsPIC33/PIC32MM MCUs
- configure the Unfied Host serial port (i.e. controller console port, not host port) Settings > Serial... Bits per second 38400 Data bits 8 Stop bits 1 Parity Disabled



[unified host serial configuration]

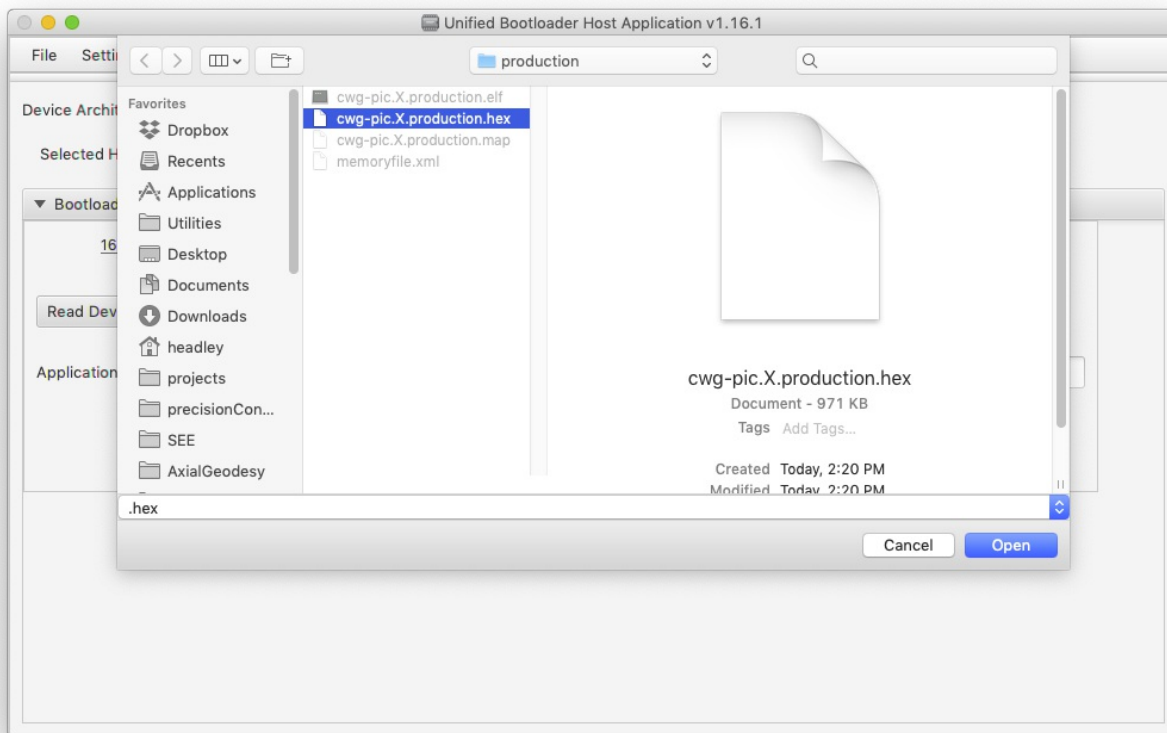
- configure the firmware image address range Application Start (Byte Address):0x1800 Application End (Byte Address):0x5B000



[unified host configuration]

- select the cwg_boot firmware image File > Open/Load (*.hex) > cwg-pic.X > dist > cwg_boot > production > cwg-

pic.X.production.hex



[select firmware hex image]

- optionally, uncheck Enable Read Back Verification After Program (slower, but recommended to verify)
- leave the Unified Host app window open
- use a terminal emulator (e.g. hyperterm, minicom, etc.) to open a console session to the controller console port - the same one used by the Unified Host. Configure using 38400N81.
- cycle power to the board
 - On power up, the LED will stay lit for 2 seconds.
 - When it turns off, type UUUU within 10 seconds to activate the bootloader
 - Each character typed should be echoed, the LED should flash when it receives a 'U'
 - The LED will stay lit when the bootloader is active (10s max or when UUUU sequence complete)
 - The LED will flash 3 times then turn off if the bootloader is **NOT** activated (cycle power and start again)
- close the terminal emulator, and return to the Unified Host session window
- click the "Program Device" button to begin programming the device. The process takes ~5-10 min.
- when the programming cycle is complete, the board should reset.
 - The LED will light for 2 S
 - After ~10s, the LED will flash three times and boot the firmware

- The LED will flash at 1 Hz for a few seconds and stop flashing if no console input is received
- Now it should be possible to initiate a terminal session on the console port (9600N81)

Test Procedures

Pre-test

- complete board prep checklist (above)
- turn off power to the controller board
- unplug all cables
- testing should be conducted on a non-conductive, static safe surface

Console and Host Port

- install console port cable
- open a terminal session (9600N81)
- issue a reset command

```
>> reset
```

```
serial IO [OK]
sdc mount [OK]
log_init [OK]
Loading config...cfg load [OK]
running boot script...
00:00:03
bootstrap [OK]

CarbonWG build [Aug 12 2020, 14:20:08]
SN: 0002
status[x0044/0000]
```

- repeat for host port, host cable
- PASS:
 - executes reset
 - both cables/ports work

Licor Serial Port

- install instrument port cable
- install loopback connector (or connect DB-9 pins 2,3)
- perform loopback test

```
>> con licor
```

[typed characters - should be echoed] * remove loopback connector

[typed characters - should **not** be echoed] * measure licor port power (pins 1,4 - ~11-12 V)

- enter CTRL-D to quit con session
- measure licor port power (pins 1,4 - ~0 V)
- PASS:
 - port power responds, voltage levels correct
 - serial port loopback test OK

Set Real Time Clock

- establish a console session
- set the time (YYYY M D H M S)

```
>> set time 2019 11 20 14 20 0
```

- get the time (should be later than set time)

```
>> get time  
2019-11-20T14:27:20Z
```

- issue reset or cycle power
- get the time (should be later than previous check)

```
>> get time  
2019-11-20T14:32:20Z
```

- PASS:
 - time retained across power cycle
 - time matches value entered

Analog Inputs

- connect 10-20k pot center/end to pins 2,3 of ANALOG port
- assert ADC enable bits

```
>> set bits a*
```

- read ADC for ends and middle of pot range

```
>> GET AD 0 1  
ADC0 : 3.5547 V  
ADC1 : 4.9805 V
```

- repeat for channels 0,1
- PASS:
 - voltage should vary X-5V ($0 < X < 5$, depending on pot value)

Battery Voltage Input

- read ADC2 (VBATT)

```
>> GET AD 2  
ADC0 : 11.4001 V
```

- PASS:
 - voltage should be ~11-12V

Valve Ports

- install 3-pin test cable on port SV0
- connect a DMM or scope to the test cable
 - DMM negative:
 - DMM positive: SVn:A
- assert all SV ports ON

```
>> set bits v* pa ps
>> get bits
```

- DMM should indicate ~11-12V
- Move the connector to each of the SV*A ports (check voltage)
- Connect the DMM to the B port and repeat for SV*B
- assert all valve ports OFF

```
>> clr bits v* pa ps
>> get bits
```

- repeat SVA, SVB checks (should be ~0.3V)
- PASS:
 - all valve ports should on/off @ nominal voltage

Pump Ports

- install 2-pin test cable on port P0
- connect a DMM or scope to the test cable
 - DMM negative: P0-
 - DMM positive: P0+
- assert all Pump ports ON

```
>> set bits p*
>> get bits
```

- DMM should indicate ~11-12V
- Move the connector to each of the pump ports (check voltage)
- assert all valve ports OFF

```
>> clr bits p*
>> get bits
```

- repeat P(checks (should be ~0.0V)
- PASS:
 - all pump ports should cycle on/off @ nominal voltage

SD Card

- run filesystem info utility

```
>>dstat
FAT type          FAT32
Bytes/Cluster     32768
Number of FATs    2
Root DIR entries  0
Sectors/FAT       3834
Number of clusters 490720
Volume start (lba) 8192
FAT start (lba)    8716
DIR start (lba,clustor) 2
Data start (lba)   16384

768312 bytes in 71 files 16 directories
15703040 KiB total 15700192 KiB available
```