MAX32660 Bootloader Code In-Application Programming with Python® User Guide

UG7159, Rev 0; 2/20

Abstract

This user guide details how update the end-user software application in the MAX32660 through the in-application programming, plus how to compile and program the MAX32660 bootloader code into the MAX32660 evaluation system (MAX32660-EVSYS). Details on the MAX32660 bootloader can be found in the **MAX32660 Bootloader User Guide**.

Table of Contents

List of Figures

List of Tables

Introduction

This application note provides the instructions to program the MAX32660 bootloader code into the MAX32660 evaluation system (MAX32660-EVSYS) and to program example host code into the MAX32630FTHR development platform.

The document also gives details related to hardware setup and application programming by using the MAX32660 bootloader and example host code. Appendices give information on compiling the MAX32666 bootloader code to evaluate the open-source MAX32666 bootloader code provided by Maxim in their own environment.

Note that the screenshots may differ according to the software versions, but the steps will be same.

System Requirements

To compile and program the MAX32660 bootloader code into the MAX32660-EVSYS, the minimum requirements are as follows:

- Windows® PC
  - Windows 10, Windows 7
  - OpenSSL
  - Maxim Toolchain Software (more information, including download and installation instructions, is in this document)
- MAX32660-EVSYS and micro-USB cable
- MAX32630FTHR and micro-USB cable
- MAX32625PICO evaluation kit (EV kit) and micro-USB cable
- Test wires to connect the MAX32660-EVSYS and MAX32630FTHR
- User should download and copy the **max32660_demo** folder inside of the release package to your PC under the *C:\maximintegrated* folder.

Maxim Toolchain Installation

To install the Maxim Toolchain to your PC, use the following steps:

1. Download the Arm® Cortex® Toolchain **here**.
2. After downloading is complete, double-click **ARMCortexToolchain.exe** and use the default settings and select **Next** until finished.
3. Select **Install the Driver/Run it Anyway** when Windows says that it does not recognize the driver.
4. In the folder *C:\Maxim*, double-click on **updates.bat**.

   If **updates.bat** fails, it may be necessary to open it in a text editor and call the commands manually.

Arm and Cortex are registered trademarks of Arm Limited.
Windows is a registered trademark of Microsoft Corporation.

MAX32660 Bootloader Code

The following step-by-step instructions provide details for programming the MAX32660 bootloader code and user binary loading in the MAX32660 through the bootloader application.

Programming the MAX32660 with the Bootloader Firmware

To program the MAX32660 bootloader code into the MAX32660-EVSYS, use the following steps:

1. Use jumper J1 to select the target VDD as 1.8V.
2. Connect the micro-USB cable to the MAX32660-EVSYS and the PC.

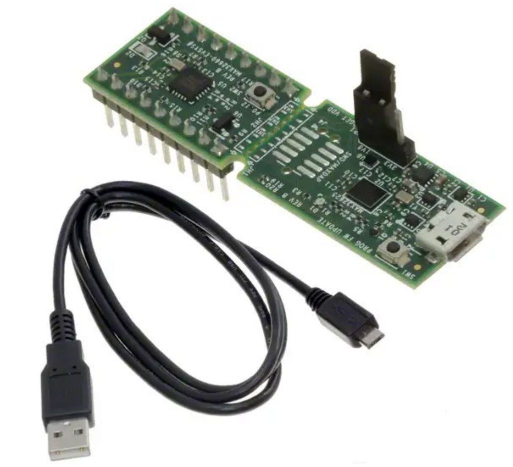


Figure 1. MAX32660-EVSYS evaluation system.

3. In the **MinGW**® window that is created after double-clicking **msys.bat** in the *C:\Maxim\Toolchain\msys\1.0* directory, navigate to the demo directory by typing in the following command, which uses forward slashes:

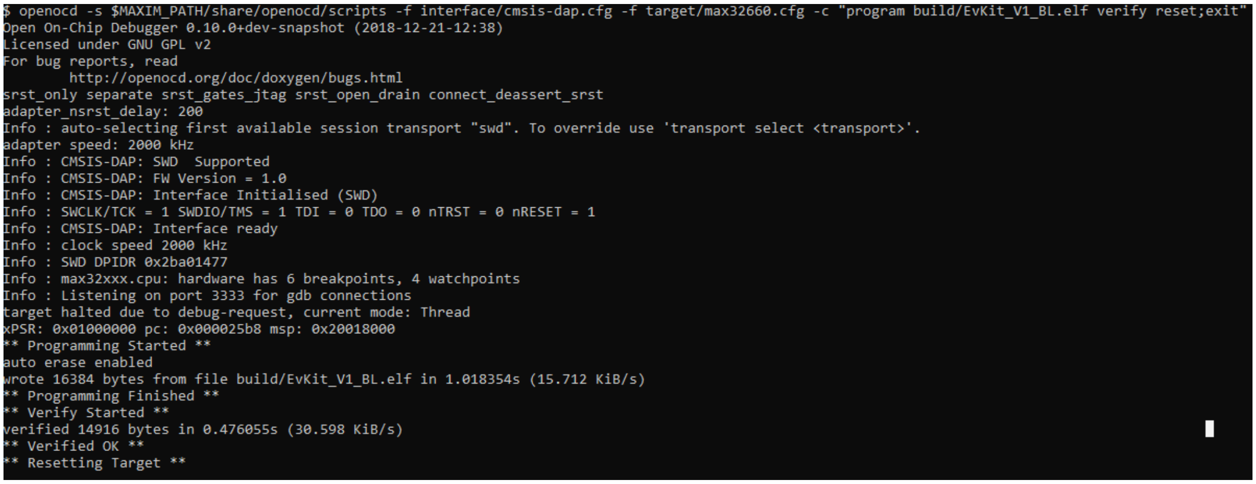
   cd c:/maximintegrated/max32660_demo

4. If your MAX32660 has been used for another version of the bootloader or for other projects, then it will be necessary to clear the flash memory of the MAX32660. In the **MinGW** window, type the following:

   openocd -s $MAXIM_PATH/share/openocd/scripts -f interface/cmsis-dap.cfg -f target/max32660.cfg -c "init;halt;max32xxx mass_erase 0;exit"

5. To flash the bootloader, type the following:

   openocd -s $MAXIM_PATH/share/openocd/scripts -f interface/cmsis-dap.cfg -f target/max32660.cfg -c "program bootloader_max32660_vx.x.x.elf verify reset;exit"



```
$ openocd -s $MAXIM_PATH/share/openocd/scripts -f interface/cmsis-dap.cfg -f target/max32660.cfg -c "program build/EvKit_V1_BL.elf verify reset;exit"
Open On-Chip Debugger 0.10.0+dev-snapshot (2018-12-21-12:38)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
srst_only separate srst_gates_jtag srst_open_drain connect_deassert_srst
adapter_nsrst_delay: 200
Info : auto-selecting first available session transport "swd". To override use 'transport select <transport>'.
adapter speed: 2000 kHz
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 1.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : clock speed 2000 kHz
Info : SWD DPIDR 0x2ba01477
Info : max32xxx.cpu: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x000025b8 msp: 0x20018000
** Programming Started **
auto erase enabled
wrote 16384 bytes from file build/EvKit_V1_BL.elf in 1.018354s (15.712 KiB/s)
** Programming Finished **
** Verify Started **
verified 14916 bytes in 0.476055s (30.598 KiB/s)
** Verified OK **
** Resetting Target **
```

Figure 2. Programming the MAX32660 bootloader.

Programming the MAX32630FTHR

To program example host code into the MAX32630FTHR, use the following steps:

1. Connect the grey 10-pin connector to the MAX32630FTHR and the MAX32625PICO board.
2. Connect the micro-USB cable to the MAX32625PICO and the PC.
3. Connect the micro-USB cable to the MAX32630FTHR and the PC.

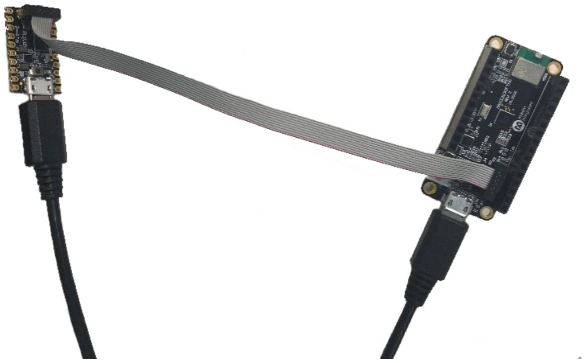
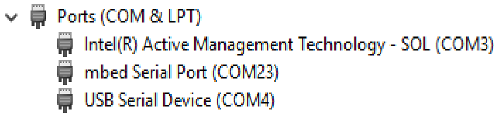


Figure 3. The MAX32630FTHR and MAX32625PICO board connection.

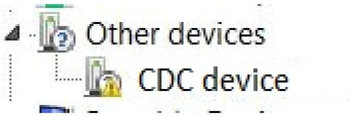
4. Wait a few minutes for the Windows driver to install, then verify that it is installed correctly.
    a. In the Windows 10 search box, type **Control Panel** (or for Windows 7, click **Control Panel** on the right side of the **Start Menu**). Either click **Hardware and Sound**, then **Device Manager**, or type **Device Manager** in the search box in the upper right.
    b. If the drivers have correctly installed, you should see one port listed as **mbed Serial Port** for the MAX32625PICO. Note the COM port number for the USB serial device.



Ports (COM & LPT)
Intel(R) Active Management Technology - SOL (COM3)
mbed Serial Port (COM23)
USB Serial Device (COM4)

Figure 4. Serial port list.

c. If you see the following, then you will need to install the correct Windows driver:



Other devices
CDC device

Figure 5. CDC device driver warning.

d. Download the Arm Mbed® Windows serial port driver **here**.

e. For Windows 10, run *mbedWinSerial_16466.exe* by double-clicking it.

f. For Windows 7,

  i. Right-click on the *mbedWinSerial_16466.exe* file, and extract to a folder.

  ii. Inside that folder, edit and add the following to the *mbedSerial_x64.inf* that the following italicized vid's and pid's are in the [_Devices] section.

```
[_Devices]
%S_DeviceDesc1%=Install,USB\VID_1F00&PID_2012&MI_01

%S_DeviceDesc1%=Install,USB\VID_1F00&PID_2012&MI_01

%S_DeviceDesc1%=Install,USB\VID_1F00&PID_2012&MI_01

%S_DeviceDesc1%=Install,USB\VID_1F00&PID_2012&REV0100

%S_DeviceDesc1%=Install,USB\VID_1F00&PID_2012
```

  iii. Right-click on the **CDC device** warning, **Update Driver Software**, **Browse My Computer** for driver software, and enter the folder name from above. Wait at least 3 to 5 minutes for the driver to install.

  iv. If there is still an issue, run the *mbed_xxx.exe* file.

5. In the **MinGW** window, navigate to the *max32660_demo* directory with the following command, which uses forward slashes:

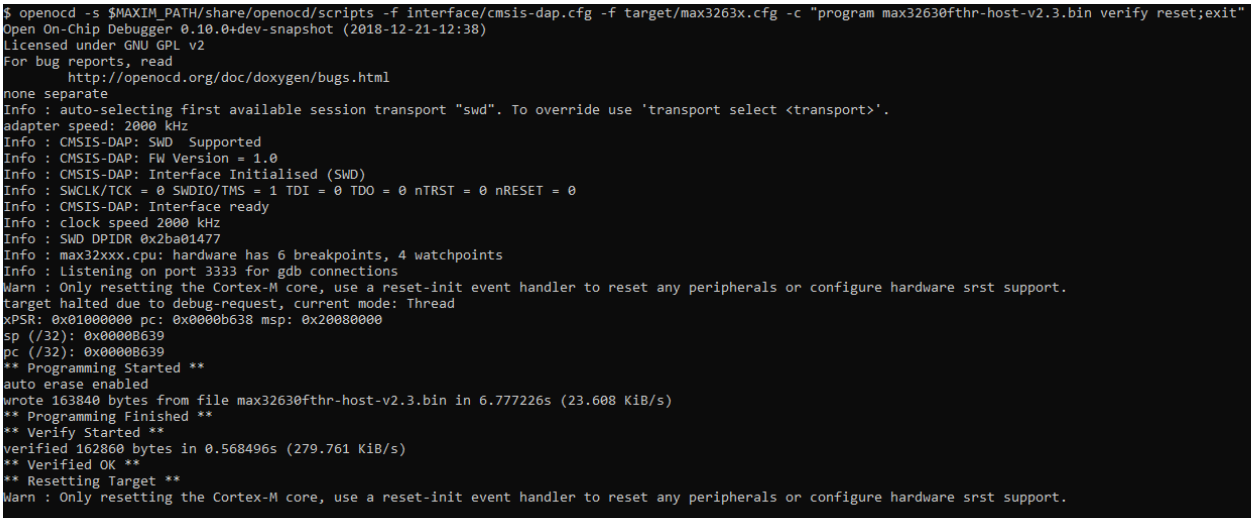
```
cd c:/maximintegrated/max32660_demo
```

6. In the **MinGW** window, type in the following:

```
openocd -s $MAXIM_PATH/share/openocd/scripts -f interface/cmsis-dap.cfg -f
target/max3263x.cfg -c "program max32630fthr-host-vx.x.bin verify reset exit”
```

Alternatively, the .bin file may be dragged and dropped into the correct DAPLINK drive.

Mbed is a registered trademark of Arm Limited.



```
$ openocd -s $MAXIM_PATH/share/openocd/scripts -f interface/cmsis-dap.cfg -f target/max3263x.cfg -c "program max32630fthr-host-v2.3.bin verify reset;exit"
Open On-Chip Debugger 0.10.0+dev-snapshot (2018-12-21-12:38)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
none separate
Info : auto-selecting first available session transport "swd". To override use 'transport select <transport>'.
adapter speed: 2000 kHz
Info : CMSIS-DAP: SWD  Supported
Info : CMSIS-DAP: FW Version = 1.0
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 0 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 0
Info : CMSIS-DAP: Interface ready
Info : clock speed 2000 kHz
Info : SWD DPIDR 0x2ba01477
Info : max32xxx.cpu: hardware has 6 breakpoints, 4 watchpoints
Info : Listening on port 3333 for gdb connections
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or configure hardware srst support.
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0000b638 msp: 0x20080000
sp (/32): 0x0000B639
pc (/32): 0x0000B639
** Programming Started **
auto erase enabled
wrote 163840 bytes from file max32630fthr-host-v2.3.bin in 6.777226s (23.608 KiB/s)
** Programming Finished **
** Verify Started **
verified 162860 bytes in 0.568496s (279.761 KiB/s)
** Verified OK **
** Resetting Target **
Warn : Only resetting the Cortex-M core, use a reset-init event handler to reset any peripherals or configure hardware srst support.
```

Figure 6. Programming the MAX32630FTHR host.

7. Press the reset button on the MAX32630FTHR, as shown in **Figure 7**.

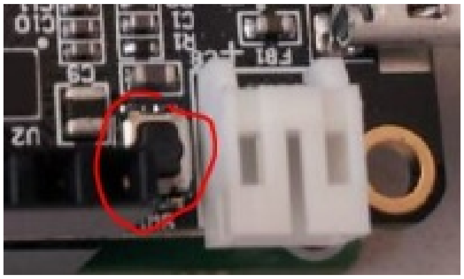


Figure 7. MAX32630FTHR host reset button.

8. Verify that the LED on the MAX32630FTHR is blinking, as shown in **Figure 8**.

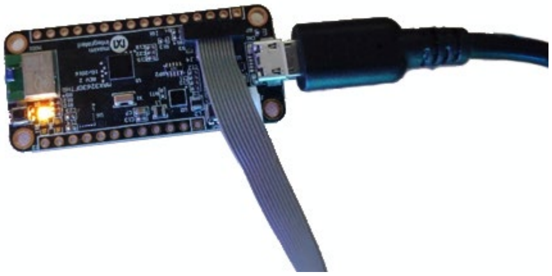


Figure 8. MAX32630FTHR host blinking LED.

Hardware Setup

Connect the MAX32630FTHR and MAX32660 with test wires according to **Table 1**. Pin diagrams for the MAX32630FTHR and MAX32660-EVSYS are given in **Figure 9** and **Figure 10**, respectively.

Table 1. Pin Connection between the MAX32630FTHR and MAX32660-EVSYS

PIN FUNCTION	MAX32660-EVSYS	MAX32630FTHR
EBL GPIO	P0.1	P5.4
I2C0_SCL	P0.2	P3.5 + 4.7K pullup
I2C0_SDA	P0.3	P3.4 + 4.7K pullup
SPI_MISO	P0.4	P5.2
SPI_MOSI	P0.5	P5.1
SPI_SCK	P0.6	P5.0
SPI_SSEL	P0.7	P5.3
UART_TX	P0.8	P3.1
UART_RX	P0.9	P3.0
RST	RSTN	P5.6

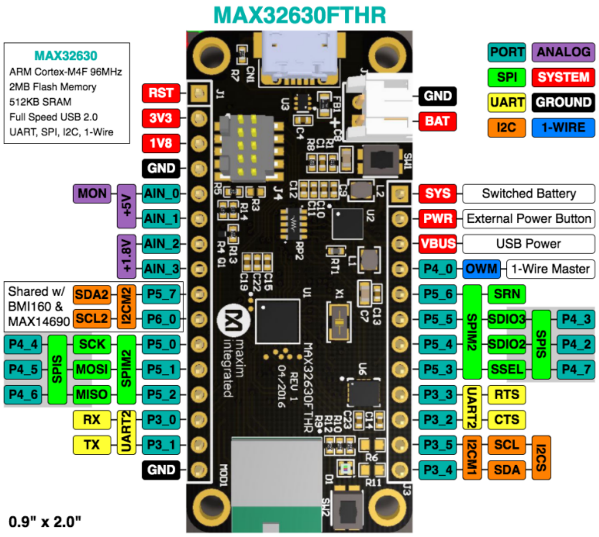
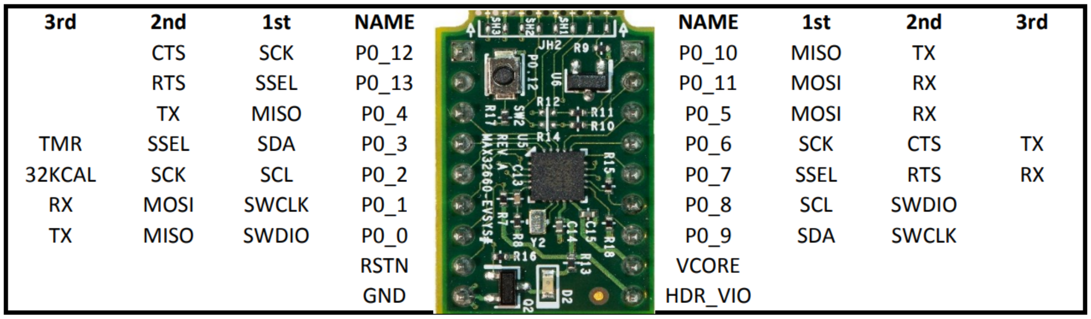


Figure 9. MAX32630FTHR pin diagram.



3rd	2nd	1st	NAME	NAME	1st	2nd	3rd
	CTS	SCK	P0_12	P0_10	MISO	TX	
	RTS	SSEL	P0_13	P0_11	MOSI	RX	
	TX	MISO	P0_4	P0_5	MOSI	RX	
TMR	SSEL	SDA	P0_3	P0_6	SCK	CTS	TX
32KCAL	SCK	SCL	P0_2	P0_7	SSEL	RTS	RX
RX	MOSI	SWCLK	P0_1	P0_8	SCL	SWDIO	
TX	MISO	SWDIO	P0_0	P0_9	SDA	SWCLK	
			RSTN	VCORE			
			GND	HDR_VIO			

Figure 10. MAX32660-EVSYS pin diagram.

In-Application Programming

Installing Python

To download and install Python, use the following steps:

1. Download and install Python 2.7.13 **here**.
2. In the Windows search box, type **dos** and select the **Command Prompt**.

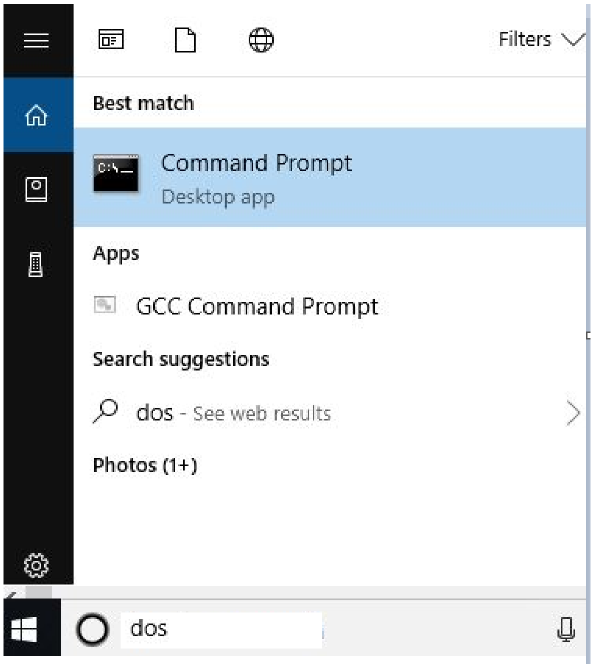


Figure 11. Navigating to the Command Prompt.

3. Navigate to this directory by typing the following at the DOS prompt:

   cd c:\maximintegrated\max32660_demo

4. Add the Python components needed by typing the following at the DOS prompt:

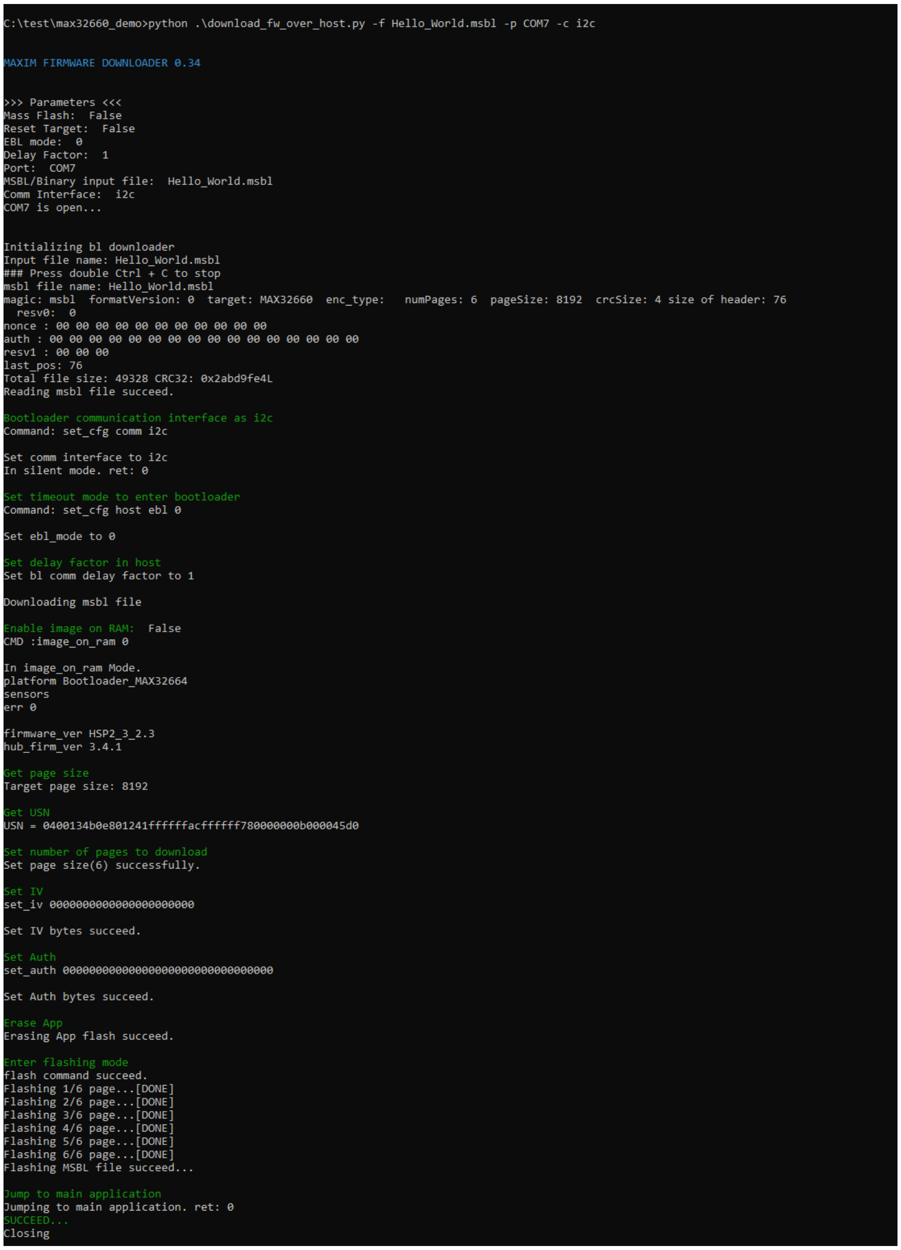
   pip install -r requirements.txt

In-Application Programming with Python

To flash the application to the MAX32660 by using the MAX32660 bootloader, use the following steps:

1. At the command prompt or in the PowerShell window, enter the following commands, replacing COMxx with the correct USB serial device COM port found in step 4b. Replace *<interface>* with the available interfaces such as uart, i2c, or spi.

   python ./download_fw_over_host.py -f "Hello_World.msbl" -p "COMxx" -d 2 -c "interface"

```
C:\test\max32660_demo>python .\download_fw_over_host.py -f Hello_World.msbl -p COM7 -c i2c


MAXIM FIRMWARE DOWNLOADER 0.34


>>> Parameters <<<
Mass Flash:  False
Reset Target:  False
EBL mode:  0
Delay Factor:  1
Port:  COM7
MSBL/Binary input file:  Hello_World.msbl
Comm Interface:  i2c
COM7 is open...


Initializing bl downloader
Input file name: Hello_World.msbl
### Press double Ctrl + C to stop
msbl file name: Hello_World.msbl
magic: msbl  formatVersion: 0  target: MAX32660  enc_type:   numPages: 6  pageSize: 8192  crcSize: 4 size of header: 76
  resv0:  0
nonce : 00 00 00 00 00 00 00 00 00 00 00
auth : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
resv1 : 00 00 00
last_pos: 76
Total file size: 49328 CRC32: 0x2abd9fe4L
Reading msbl file succeed.

Bootloader communication interface as i2c
Command: set_cfg comm i2c

Set comm interface to i2c
In silent mode. ret: 0

Set timeout mode to enter bootloader
Command: set_cfg host ebl 0

Set ebl_mode to 0

Set delay factor in host
Set bl comm delay factor to 1

Downloading msbl file

Enable image on RAM:  False
CMD :image_on_ram 0

In image_on_ram Mode.
platform Bootloader_MAX32664
sensors
err 0

firmware_ver HSP2_3_2.3
hub_firm_ver 3.4.1

Get page size
Target page size: 8192

Get USN
USN = 0400134b0e801241ffffffacffffff780000000b000045d0

Set number of pages to download
Set page size(6) successfully.

Set IV
set_iv 0000000000000000000000

Set IV bytes succeed.

Set Auth
set_auth 00000000000000000000000000000000

Set Auth bytes succeed.

Erase App
Erasing App flash succeed.

Enter flashing mode
flash command succeed.
Flashing 1/6 page...[DONE]
Flashing 2/6 page...[DONE]
Flashing 3/6 page...[DONE]
Flashing 4/6 page...[DONE]
Flashing 5/6 page...[DONE]
Flashing 6/6 page...[DONE]
Flashing MSBL file succeed...

Jump to main application
Jumping to main application. ret: 0
SUCCEED...
Closing
```

   Figure 12. Downloading firmware with the download_fw_over_host Python script.

2. The application has now been flashed to the chip. For advanced usage of flasher script, refer to the *Firmware_downloader_usage.txt* file.

Appendix A: Compiling the Bootloader Code with the Make Command

To compile the MAX32660 bootloader code with the make command, use the following steps:

1. Navigate to the max32660_bootloader_src directory in the **MinGW** window with the following command, which uses forward slashes:

   cd c:/maximintegrated/max32660_bootloader_src

2. Enter the following command in the **MinGW** window and wait several minutes for the command to complete:

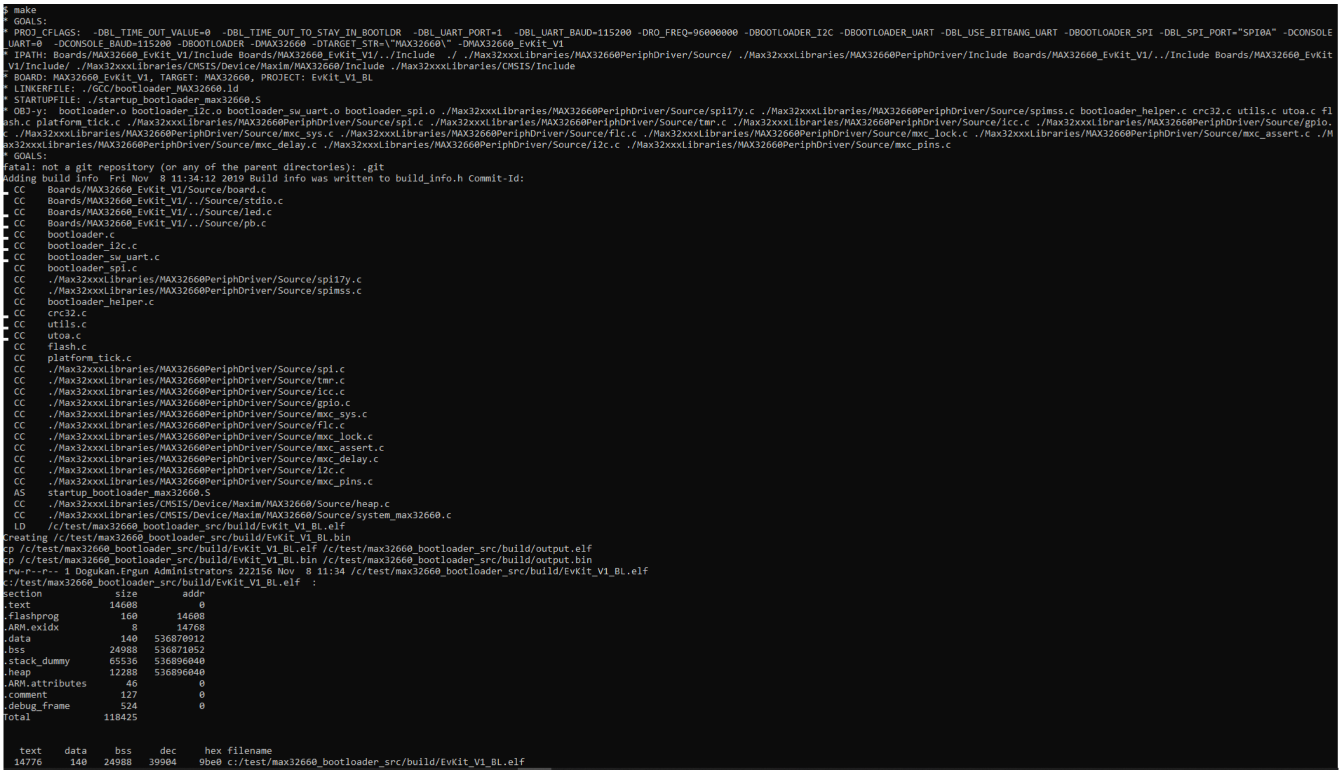
   make

3. After successful compiling, the *EvKit_V1_BL.elf* bootloader executable will be in the *C:\maximintegrated\max32660_bootloader_src\build* directory.

   If you want to rebuild, then enter these commands respectively:

   make clean

   make



```
$ make
* GOALS:
* PROJ_CFLAGS:  -DBL_TIME_OUT_VALUE=0  -DBL_TIME_OUT_TO_STAY_IN_BOOTLDR  -DBL_UART_PORT=1  -DBL_UART_BAUD=115200 -DRO_FREQ=96000000 -DBOOTLOADER_I2C -DBOOTLOADER_UART -DBL_USE_BITBANG_UART -DBOOTLOADER_SPI -DBL_SPI_PORT="SPI0A" -DCONSOLE
_UART=0  -DCONSOLE_BAUD=115200 -DBOOTLOADER -DMAX32660 -DTARGET_STR=\"MAX32660\" -DMAX32660_EvKit_V1
* IPATH: Boards/MAX32660_EvKit_V1/Include Boards/MAX32660_EvKit_V1/../Include  ./ ./Max32xxxLibraries/MAX32660PeriphDriver/Source/ ./Max32xxxLibraries/MAX32660PeriphDriver/Include Boards/MAX32660_EvKit_V1/../Include Boards/MAX32660_EvKit
_V1/Include/ ./Max32xxxLibraries/CMSIS/Device/Maxim/MAX32660/Include ./Max32xxxLibraries/CMSIS/Include
* BOARD: MAX32660_EvKit_V1, TARGET: MAX32660, PROJECT: EvKit_V1_BL
* LINKERFILE: ./GCC/bootloader_MAX32660.ld
* STARTUPFILE: ./startup_bootloader_max32660.S
* OBJ-y:  bootloader.o bootloader_i2c.o bootloader_sw_uart.o bootloader_spi.o ./Max32xxxLibraries/MAX32660PeriphDriver/Source/spi17y.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/spimss.c bootloader_helper.c crc32.c utils.c utoa.c fl
ash.c platform_tick.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/spi.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/tmr.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/icc.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/gpio.
c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_sys.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/flc.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_lock.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_assert.c ./M
ax32xxxLibraries/MAX32660PeriphDriver/Source/mxc_delay.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/i2c.c ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_pins.c
* GOALS:
fatal: not a git repository (or any of the parent directories): .git
Adding build info  Fri Nov  8 11:34:12 2019 Build info was written to build_info.h Commit-Id:
  CC    Boards/MAX32660_EvKit_V1/Source/board.c
  CC    Boards/MAX32660_EvKit_V1/../Source/stdio.c
  CC    Boards/MAX32660_EvKit_V1/../Source/led.c
  CC    Boards/MAX32660_EvKit_V1/../Source/pb.c
  CC    bootloader.c
  CC    bootloader_i2c.c
  CC    bootloader_sw_uart.c
  CC    bootloader_spi.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/spi17y.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/spimss.c
  CC    bootloader_helper.c
  CC    crc32.c
  CC    utils.c
  CC    utoa.c
  CC    flash.c
  CC    platform_tick.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/spi.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/tmr.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/icc.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/gpio.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_sys.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/flc.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_lock.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_assert.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_delay.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/i2c.c
  CC    ./Max32xxxLibraries/MAX32660PeriphDriver/Source/mxc_pins.c
  AS    startup_bootloader_max32660.S
  CC    ./Max32xxxLibraries/CMSIS/Device/Maxim/MAX32660/Source/heap.c
  CC    ./Max32xxxLibraries/CMSIS/Device/Maxim/MAX32660/Source/system_max32660.c
  LD    /c/test/max32660_bootloader_src/build/EvKit_V1_BL.elf
Creating /c/test/max32660_bootloader_src/build/EvKit_V1_BL.bin
cp /c/test/max32660_bootloader_src/build/EvKit_V1_BL.elf /c/test/max32660_bootloader_src/build/output.elf
cp /c/test/max32660_bootloader_src/build/EvKit_V1_BL.bin /c/test/max32660_bootloader_src/build/output.bin
-rw-r--r-- 1 Dogukan.Ergun Administrators 222156 Nov  8 11:34 /c/test/max32660_bootloader_src/build/EvKit_V1_BL.elf
c:/test/max32660_bootloader_src/build/EvKit_V1_BL.elf  :
section             size        addr
.text              14608           0
.flashprog           160       14608
.ARM.exidx             8       14768
.data                140   536870912
.bss               24988   536871052
.stack_dummy       65536   536896040
.heap              12288   536896040
.ARM.attributes       46           0
.comment             127           0
.debug_frame         524           0
Total             118425


   text    data     bss     dec     hex filename
  14776     140   24988   39904    9be0 c:/test/max32660_bootloader_src/build/EvKit_V1_BL.elf
```

Figure 13. Compiling bootloader code in the MinGW window.

Appendix B: Converting .bin file to .msbl file

Convert the .bin application program to an .msbl file by entering the following command in the DOS window to program by using the bootloader:

```
msblGenWin32.exe myapplication.bin MAX32660 8192
```

Be sure that you have used correct linker file at your application. A sample linker file (*max32660.ld*) can be found at *max32660_bootloader_src\example\Hello_World*.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	2/20	Initial release	—