

Updating Arduino 101 Firmware

User Guide

March 2016



Table of Contents

1	Introduction	3
1.1	Prerequisites	3
2	Installation and Setup	4
2.1	Downloading the source code	4
2.2	Installing prerequisite packages	4
2.3	Building images	4
2.4	Creating flashpack.zip	4
2.4.1	Windows	5
2.4.2	Linux/Mac	6

Tables

Figure 1	Creating flashback.zip	5
Figure 2	Flashing images to the board with Windows– open command window	6
Figure 3	Flashing images to the board with Windows – execute command	6
Figure 4	Flashing images to the board with Linux/Mac – reset the board	6
Figure 5	Flashing images to the board with Linux/Mac – successful flash	7



1 Introduction

1.1 Prerequisites

This firmware is intended to be built only on 64 bit Ubuntu systems. If your native machine is not running a 64 bit Ubuntu operating system, you can still perform the firmware building process using an Ubuntu 14 64 bit OS in a virtual machine with 15GB of HDD space allocated. It is advised that you do not use a live USB or CD. Doing so will result in some steps failing, as it would not be possible to download the required packages.

If you have not configured the Arduino101 board yet, follow the steps in the link below before downloading the source code. Verify that you can successfully run a blink sketch to ensure the DFU is operating correctly. This is important, as it will provide you with the platform to flash the binaries.

<https://www.arduino.cc/en/Guide/Arduino101>





2 Installation and Setup

2.1 Downloading the source code

Visit <https://downloadcenter.intel.com/download/25832> and download the tar-ball located there. When the download is complete, move it into the folder where the tar-ball is saved, and execute the following commands:

```
$ tar -xf arduino101_firmware_source-v1.tar.bz2
$ cd arduino101_firmware_source-v1
$ project_directory=$(pwd)/arduino101_firmware/projects/arduino101
```

2.2 Installing prerequisite packages

Verify that packages from all repositories are up to date:

```
$ sudo apt-get update
```

Ensure that you have all the required packages before compiling. As the target suggests, this is the only required the first time you compile:

```
$ sudo make one_time_setup -C $project_directory
```

This installs the following packages:

```
gawk wget git-core diffstat unzip texinfo
gcc-multilib build-essential chrpath libsdl1.2-dev xterm
libqtgui4:i386 libtool libc6:i386
libc6-dev-i386 autoconf libtool pkg-config gperf flex bison
```

2.3 Building images

Enter this command each time you modify the code to update the images:

```
$ make clean setup image -C $project_directory
```

2.4 Creating flashpack.zip

This command creates "flashpack.zip" which is used for flashing the board:

```
$ arduino101_flashpack/create_flasher.sh
```

Figure 1 Creating flashback.zip

```
building file list ... done
created directory images/firmware
./
Files
bootloader_quark.bin -> /home/test/v1/out/current/quark_se/quark/bootloader/bootlo
bootupdater.bin
quark.0.bin
quark.1.bin
quark.bin
quark.padded.bin

sent 386,860 bytes  received 193 bytes  258,035.33 bytes/sec
total size is 386,202  speedup is 1.00
adding: arduino101_flashpack/ (stored 0%)
adding: arduino101_flashpack/drivers/ (stored 0%)
adding: arduino101_flashpack/drivers/rules.d/ (stored 0%)
adding: arduino101_flashpack/drivers/rules.d/99-dfu.rules (deflated 40%)
adding: arduino101_flashpack/images/ (stored 0%)
adding: arduino101_flashpack/images/firmware/ (stored 0%)
adding: arduino101_flashpack/images/firmware/quark.bin (deflated 47%)
adding: arduino101_flashpack/images/firmware/FSRom.bin (deflated 97%)
adding: arduino101_flashpack/images/firmware/quark.1.bin (deflated 100%)
adding: arduino101_flashpack/images/firmware/arc.bin (deflated 41%)
adding: arduino101_flashpack/images/firmware/bootloader_quark.bin (deflated 46%)
adding: arduino101_flashpack/images/firmware/quark.0.bin (deflated 79%)
adding: arduino101_flashpack/images/firmware/quark.padded.bin (deflated 81%)
adding: arduino101_flashpack/images/firmware/bootupdater.bin (deflated 54%)
adding: arduino101_flashpack/utilities/ (stored 0%)
adding: arduino101_flashpack/utilities/fwversion/ (stored 0%)
adding: arduino101_flashpack/utilities/fwversion/readbin.c (deflated 41%)
adding: arduino101_flashpack/utilities/fwversion/README (deflated 7%)
adding: arduino101_flashpack/fwversion.bat (deflated 36%)
adding: arduino101_flashpack/bin/ (stored 0%)
adding: arduino101_flashpack/bin/LICENSE.dfu-util (deflated 62%)
adding: arduino101_flashpack/bin/readbin (deflated 69%)
adding: arduino101_flashpack/bin/libwinpthread-1.dll (deflated 54%)
adding: arduino101_flashpack/bin/dfu-util.32 (deflated 54%)
adding: arduino101_flashpack/bin/readbin.exe (deflated 60%)
adding: arduino101_flashpack/bin/readbin.32 (deflated 64%)
adding: arduino101_flashpack/bin/dfu-util.exe (deflated 66%)
adding: arduino101_flashpack/bin/dfu-util (deflated 61%)
adding: arduino101_flashpack/flash_dfu.bat (deflated 58%)
adding: arduino101_flashpack/create_flasher.sh (deflated 48%)
adding: arduino101_flashpack/bin_osx/ (stored 0%)
adding: arduino101_flashpack/bin_osx/dfu-prefix (deflated 66%)
adding: arduino101_flashpack/bin_osx/LICENSE.dfu-util (deflated 62%)
adding: arduino101_flashpack/bin_osx/readbin (deflated 66%)
adding: arduino101_flashpack/bin_osx/LICENSE.libusb (deflated 65%)
adding: arduino101_flashpack/bin_osx/dfu-util (deflated 58%)
adding: arduino101_flashpack/bin_osx/dfu-suffix (deflated 66%)
adding: arduino101_flashpack/bin_osx/LICENSE (deflated 66%)
adding: arduino101_flashpack/fwversion.sh (deflated 44%)
adding: arduino101_flashpack/flash_dfu.sh (deflated 55%)
```

2.5 Flashing images to the board

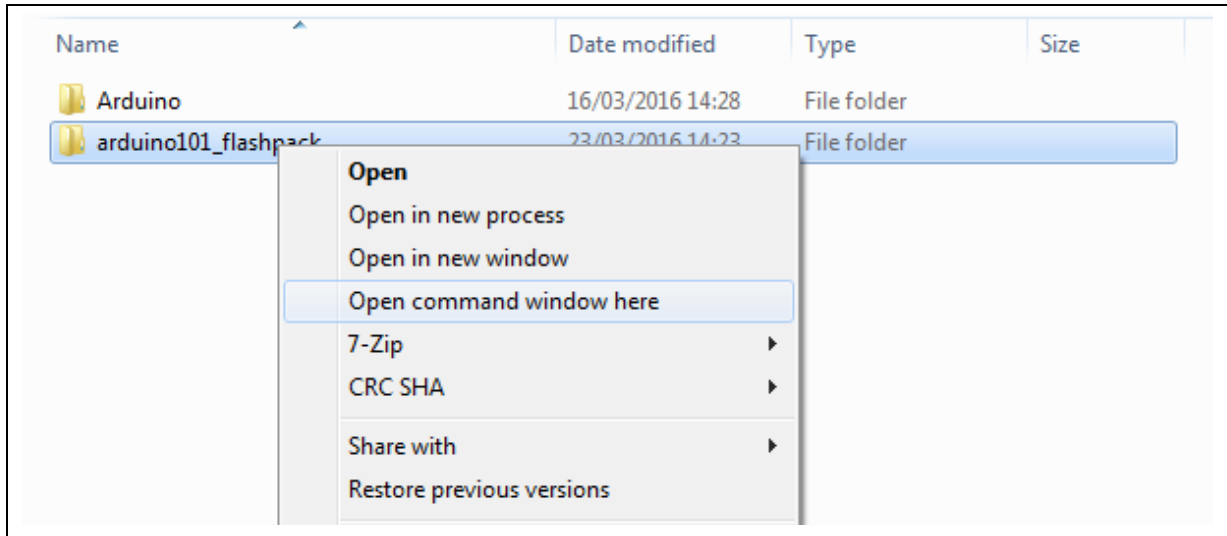
Ensure the board is connected via USB, move the .zip file to the machine where the Arduino IDE is installed, and extract there.

2.5.1 Windows

Shift+Ctrl+right click mouse on extracted folder and click "Open command window here".



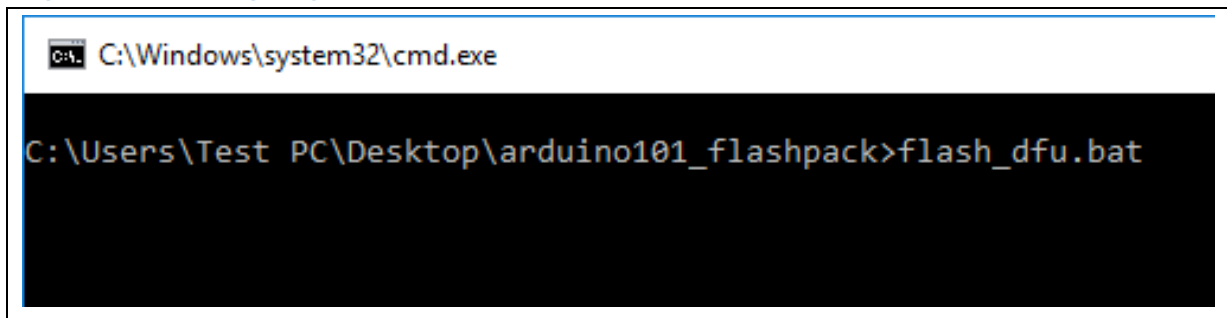
Figure 2 Flashing images to the board with Windows– open command window



Execute command:

```
flash_dfu.bat
```

Figure 3 Flashing images to the board with Windows – execute command

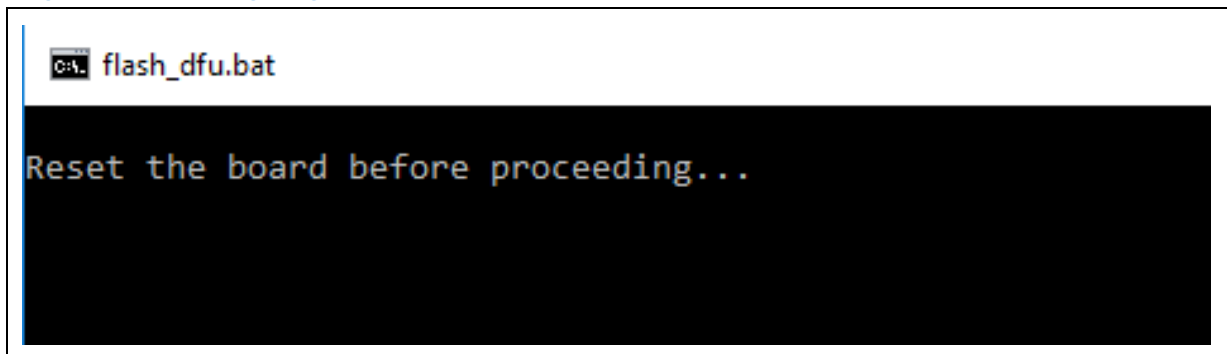


2.5.2 Linux/Mac

```
$ cd arduino101_flashpack  
$ ./flash_dfu.sh
```

Press the reset button on the board to begin the flash process.

Figure 4 Flashing images to the board with Linux/Mac – reset the board



The figure below shows an example of a successful flash.

Figure 5 Flashing images to the board with Linux/Mac – successful flash

```
C:\> Select flash_dfu.bat

Please report bugs to dfu-util@lists.gnumonks.org

Invalid DFU suffix signature
A valid DFU suffix will be required in a future dfu-util release!!!
Opening DFU capable USB device...
ID 8087:0aba
Run-time device DFU version 0011
Claiming USB DFU Interface...
Setting Alternate Setting #2 ...
Determining device status: state = dfuIDLE, status = 0
dfuIDLE, continuing
DFU mode device DFU version 0011
Device returned transfer size 2048
Copying data from PC to DFU device
Download      [=====] 100%          51680 bytes
Download done.
state(2) = dfuIDLE, status(0) = No error condition is present
Done!
dfu-util 0.8

Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.
Copyright 2010-2014 Tormod Volden and Stefan Schmidt
This program is Free Software and has ABSOLUTELY NO WARRANTY
Please report bugs to dfu-util@lists.gnumonks.org

Invalid DFU suffix signature
A valid DFU suffix will be required in a future dfu-util release!!!
Opening DFU capable USB device...
ID 8087:0aba
Run-time device DFU version 0011
Claiming USB DFU Interface...
Setting Alternate Setting #7 ...
Determining device status: state = dfuIDLE, status = 0
dfuIDLE, continuing
DFU mode device DFU version 0011
Device returned transfer size 2048
Copying data from PC to DFU device
Download      [===== ] 89%          30720 bytesEr
can't detach
Resetting USB to switch back to runtime mode

---SUCCESS---
Press any key to continue . . .
```