

Purpose of the script

The script installs the ArCoMoRa software on an Arduino connected with USB n.

How the script works

The script does a few things

1. Asks the user for the device (the Arduino) to be flashed
2. The script asks the user which ArCoMoRa software to install. That can be:
 - a. Mardec
 - b. ArLoCo
 - c. ArSigDec

Of these three versions are available for an Uno, Mega or a Micro.

The script asks which version to install.

3. Asks for confirmation and then starts flashing.

Technical explanation

The script has as input compiled hex files that are suitable for an Arduino. Flashing uses package avrdude with an avrdude.config included in the ZIP file.

Preparations

1. The script is delivered in a ZIP file together with the ArCoMoRa hex files. These files must stay together. Preferably put them in a separate folder.
2. The script must be made executable with the command: `chmod +x uploadV12_EN.sh`.
3. Package avrdude must be installed. Do this with the package manager (Apt, Pacman, Portage, etc.). The script aborts if avrdude is not present..

```
Error: the required package avrdude is not installed.  
Install this package first. The installation method is different for each Linux distrubution.  
The script will now exit.
```

4. The user running the script must have write access to the USB port to which the Arduino is connected. This can be arranged in several ways, but the most common and neat way is to add the user to the same user group that already has write access to the USB port.
The script aborts if it doesn't:

```
Error: You have no write access to the selected device  
Add yourself to the group dialout  
The script will now exit.
```

Step by step instructions

1. Start the script with `./uploadv12_EN.sh`

```
Welcome to the ArCoMoRA upload script. With this script you can flash Arcomora software to an Arduino/DCCNext
In a few steps some checks are made and you must select the device and the program to flash.
Script version 1.1 - script made by erikkral.
```

```
Step 1: Select an USB Serial device:
```

- ```
1) /dev/ttyUSB0
2) Stop
```

```
Enter the number of the device:
```

2. Select the device to be flashed. In this example only one is connected.  
This makes it easier to select the device.

```
Step 2: Choose an option to flash:
```

- ```
1) Mardec for Uno/DCCNext      4) ArLoco for Mega          7) ArSigDec for Mega
2) Mardec for Mega             5) ArLoco for Nano          8) ArSigDec for Nano
3) ArLoco for Uno              6) ArSigDec for Uno/DCCNext 9) Stop
```

3. Select the file to be flashed. Enter the number and press enter.

```
Enter the number:1
```

```
Selected: Mardec for Uno/DCCNext.
```

```
Step 3: The flashing can start now. Confirm to continue or stop.
```

- ```
1) Continue
2) Stop
```

```
Continue or stop:
```

4. The script asks for confirmation. After confirmation, the flashing starts:

```
Continue or stop:1
```

```
Step 4: Now flashing file. avrdude will be started
Look at the screen for possible errors.
```

```
avrdude: Version 6.3-20171130
```

```
Copyright (c) 2000-2005 Brian Dean, http://www.bdmicro.com/
```

```
Copyright (c) 2007-2014 Joerg Wunsch
```

```
System wide configuration file is "avrdude.local"
```

```
User configuration file is "/home/erik/.avrduderc"
```

```
User configuration file does not exist or is not a regular file, skipping
```

```
Using Port : /dev/ttyUSB0
```

```
Using Programmer : arduino
```

```
Overriding Baud Rate : 115200
```

```

PollIndex : 3
PollValue : 0x53
Memory Detail :

```

| Memory      | Type | Mode | Delay | Block Size | Poll Indx | Paged | Size  | Page Size | #Pages | MinW | MaxW | Polled ReadBack |
|-------------|------|------|-------|------------|-----------|-------|-------|-----------|--------|------|------|-----------------|
| eeptom      |      | 65   | 20    | 4          | 0         | no    | 1024  | 4         | 0      | 3600 | 3600 | 0xff 0xff       |
| flash       |      | 65   | 6     | 128        | 0         | yes   | 32768 | 128       | 256    | 4500 | 4500 | 0xff 0xff       |
| lfuse       |      | 0    | 0     | 0          | 0         | no    | 1     | 0         | 0      | 4500 | 4500 | 0x00 0x00       |
| hfuse       |      | 0    | 0     | 0          | 0         | no    | 1     | 0         | 0      | 4500 | 4500 | 0x00 0x00       |
| efuse       |      | 0    | 0     | 0          | 0         | no    | 1     | 0         | 0      | 4500 | 4500 | 0x00 0x00       |
| lock        |      | 0    | 0     | 0          | 0         | no    | 1     | 0         | 0      | 4500 | 4500 | 0x00 0x00       |
| calibration |      | 0    | 0     | 0          | 0         | no    | 1     | 0         | 0      | 0    | 0    | 0x00 0x00       |
| signature   |      | 0    | 0     | 0          | 0         | no    | 3     | 0         | 0      | 0    | 0    | 0x00 0x00       |

```

Programmer Type : Arduino
Description : Arduino
Hardware Version: 3
Firmware Version: 4.4
Vtarget : 0.3 V
Varef : 0.3 V
Oscillator : 28.800 kHz
SCK period : 3.3 us

```

```

avrdude: AVR device initialized and ready to accept instructions

Reading | ##### | 100% 0.00s

avrdude: Device signature = 0x1e950f (probably m328p)
avrdude: safemode: lfuse reads as 0
avrdude: safemode: hfuse reads as 0
avrdude: safemode: efuse reads as 0
avrdude: reading input file "MARDEC.hex"
avrdude: writing flash (31698 bytes):

Writing | ##### | 100% 5.26s

avrdude: 31698 bytes of flash written
avrdude: verifying flash memory against MARDEC.hex:
avrdude: load data flash data from input file MARDEC.hex:
avrdude: input file MARDEC.hex contains 31698 bytes
avrdude: reading on-chip flash data:

Reading | ##### | 100% 4.06s

avrdude: verifying ...
avrdude: 31698 bytes of flash verified

avrdude: safemode: lfuse reads as 0
avrdude: safemode: hfuse reads as 0
avrdude: safemode: efuse reads as 0
avrdude: safemode: Fuses OK (E:00, H:00, L:00)

avrdude done. Thank you.

avrdude has finished flashing
Check the screen for possible errors

The script will now exit.

```

- The output is quite difficult to understand. Above is an example of a successful attempt to flash. When flashing the output must look like the screens above.  
The 3 counters must have reached 100% and no other error messages should be displayed at the end.

Flashing is now successful!