The GRUB Switch - Quickstart Guide

Ruediger Willenberg
Version 1.9
May 13, 2023

Note: *The GRUB Switch* only works with *GRUB 2.x*, not the outdated *GRUB Legacy*

This guide expects that you know how to use the Linux command-line in a console window and are familiar with commands to change directories, list files and start bash scripts. You will also need *super user* access for some steps, so make sure you have **`sudo`** rights.

Refer to the detailed documentation if these quick steps do not work on your platform.

Table of Contents

1 Preliminary steps to take for GRUB v2.06 and higher

Starting with GRUB version 2.06 which was released in 2022, Windows and Linux installations on other partitions are not guaranteed to be included in the GRUB boot menu.

To find out if GRUB v.2.06 or higher is installed, call

```
grub-install --version
```

If the version is 2.06 or higher, we need to modify the parameter file **`/etc/default/grub`** that controls how the GRUB menu is assembled. Open the file with a text editor with *root* rights, for example with

```
sudo nano /etc/default/grub
```

Add the line

```
GRUB_DISABLE_OS_PROBER=false
```

and save the file. Afterwards, run **`update-grub`** at least once to rebuild the GRUB menu with

```
sudo update-grub
```

before using our *GRUB Switch* command-line tools as detailed below.

2 Downloading and installing GRUB Switch repository

The GRUB Switch scripts and files are designed to be installed and used under Linux. There are two ways to obtain the files:

- Via **git clone**:
  - Go to your preferred working directory and clone the repository with

    ```
    git clone  https://github.com/rw-hsma-fpga/grub-switch.git
    ```

  - All GRUB Switch files are now available in the path

    ```
    ./grub-switch
    ```

- Via **download**:
  - Download to your preferred directory: https://github.com/rw-hsma-fpga/grub-switch/archive/refs/heads/master.zip
  - Unzip with GUI-tool or on the command line with

    ```
    unzip grub-switch-master.zip
    ```

  - All GRUB Switch files are now available in the path

    ```
    ./grub-switch-master
    ```

All further examples will assume you are starting in the parent directory above **`grub-switch`**

3 Configuring your boot choices

To configure your *GRUB Switch* boot choices, enter the bash script directory with

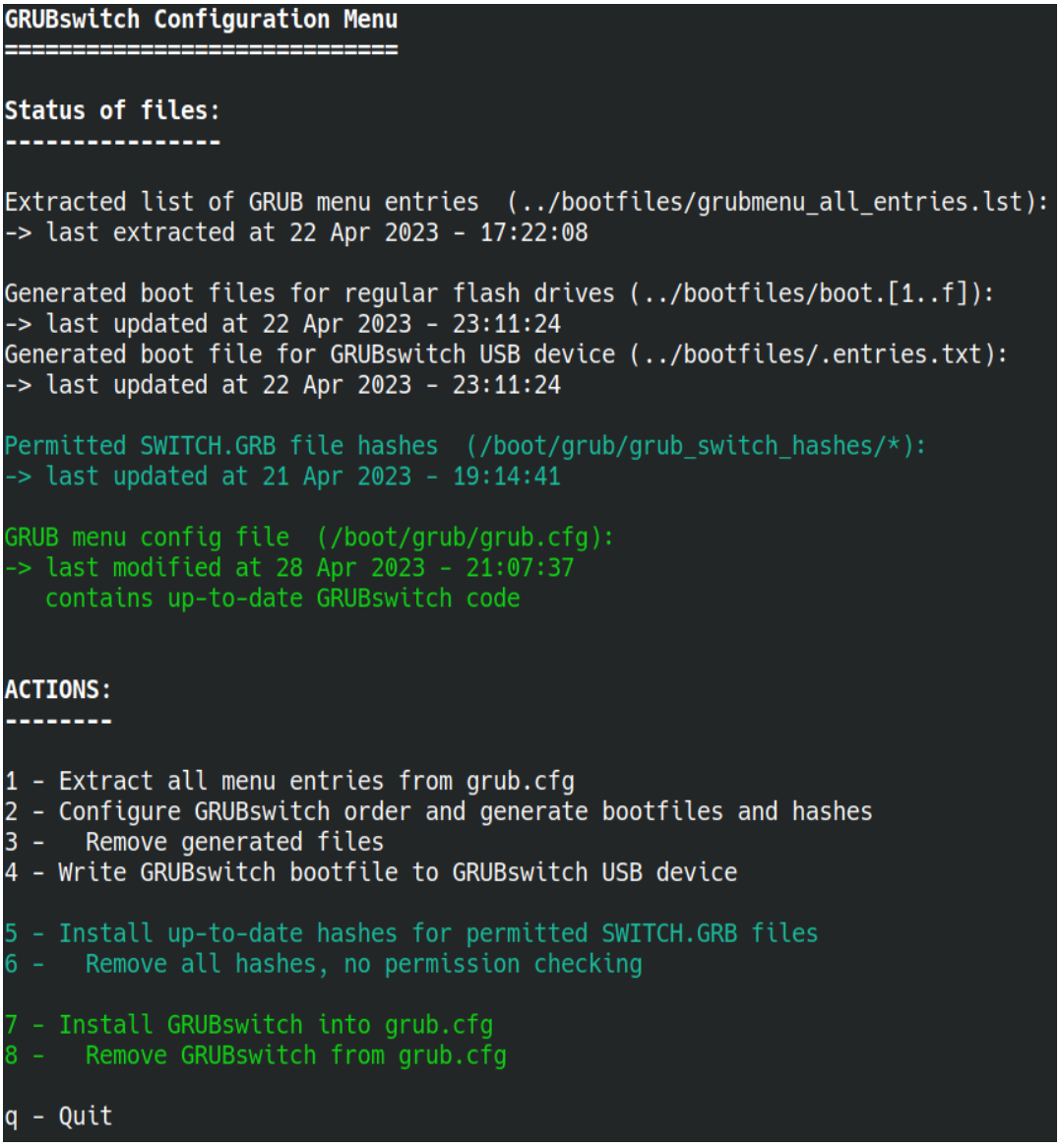
```
cd grub-switch/1_shell_scripts/
```

(exact path depends on your installation location and current working directory)

Start the configuration tool with

```
./CONFIGURE_GRUBswitch.sh
```

Do not start the configuration tool with *sudo* or all generated files will be owned by *root*. The script asks for *sudo* once right after you start it, and uses it when super user rights are required. You may be asked again when *sudo* privileges time out.



The main screen then should appear as above (depending on your console theme).

Below that, the script lists the status of important files either required or generated by this tool.

In the lower half, the tool offers a menu of actions to take by pressing a corresponding number key.

3.1 Extracting a list of GRUB menu entries

By choosing option **1**, the tool will parse GRUB's menu configuration file and generate a local text file **`grub-switch/bootfiles/grubmenu_all_entries.txt`** that lists all menu entries, including ones in submenus, and is the basis for picking your custom boot choices.

Before the extraction, you have the option to re-generate the **`grub.cfg`** with **`update-grub`** so that your extracted choices reflect the most recent boot options. This is generally recommended, with the exception of *Linux Mint* (see section on specific Linux distributions).

After returning to the main screen, the status of **`grubmenu_all_entries.txt`** should reflect the recent extraction date.

3.2 Selecting and ordering your boot choices

```
* Use Cursor Up / Cursor Down / Pos1 / End keys to navigate
* Assign a GRUB Switch choice position to an entry
  by pressing the keys 1..9 or a..f(=10..15)
  [The 0 position is reserved for the GRUB Menu]
* Press Delete to remove choice position from current entry
* Press Insert to assign/remove all positions in order
* Press q to quit
* Press Enter to continue
------------------------------------------------------------
 POS | ENTRY
-----|-------
  .  | Ubuntu
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-73-generic
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-73-generic (recovery mode)
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-72-generic
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-72-generic (recovery mode)
  .  | Windows Boot Manager (on /dev/nvme0n1p1)
  .  | System setup
```

Choose menu option **2** to start the actual *GRUB Switch* configuration.

The tool will show a complete list of boot choices in the order extracted from the GRUB configuration. Moving up and down with the cursor keys, you can use number keys (and the hexadecimal letters **a** to **f** representing 10..15) to assign choices a position in the *GRUB Switch* selection order.

```
* Use Cursor Up / Cursor Down / Pos1 / End keys to navigate
* Assign a GRUB Switch choice position to an entry
  by pressing the keys 1..9 or a..f(=10..15)
  [The 0 position is reserved for the GRUB Menu]
* Press Delete to remove choice position from current entry
* Press Insert to assign/remove all positions in order
* Press q to quit
* Press Enter to continue
------------------------------------------------------------
 POS | ENTRY
-----|-------
  2  | Ubuntu
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-73-generic
  3  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-73-generic (recovery mode)
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-72-generic
  .  | Advanced options for Ubuntu > Ubuntu, with Linux 5.4.0-72-generic (recovery mode)
  1  | Windows Boot Manager (on /dev/nvme0n1p1)
  4  | System setup
```

Not all entries have to be chosen and assigned a numerical position. The *Enter* key continues to the next screen.

```
Switch positions chosen assigned:
---------------------------------
0 : GRUB Menu (Fixed)
1 : Windows Boot Manager (on /dev/nvme0n1p1)
2 : Ubuntu
3 : Advanced options for Ubuntu>Ubuntu, with Linux 5.4.0-73-generic (recovery mode)
4 : System setup
5 :
6 :
7 :
8 :
9 :
a :
b :
c :
d :
e :
f :
-----------------------------------------
* Press Enter to confirm selection
* Press Backspace <- to change configuration
* Press q to quit
```

The next screen shows the selected choices and position assignments. They can be confirmed with the *Enter* key; the *Backspace* key returns to the previous screen for further changes.

```
------------------------------------------------------
Set boot choice display time (min 2 secs) :
* Press Cursor Up / Cursor Down to change +/- 10 seconds
* Press Cursor Right/ Cursor Left to change +/- 1 second
* Press Enter to confirm selection
* Press q to quit without changes
------------------------------------------------------

Show boot choice for 005 seconds
```

Before booting, the chosen boot option will be displayed for a number of seconds, which can be adjusted in the next tool screen. The minimum selectable time is 2 seconds, so that there will always be a chance to reach the GRUB menu with the *Escape* key.

```
------------------------------------------------------
Choose highlight colors for boot choice display:
* Press Cursor Right/ Cursor Left to change
* Press Enter to confirm selection
* Press Backspace <- to go back and change display time
* Press q to quit
------------------------------------------------------

Current choice is white/red.
See example below:


   Booting FluxCapOS 1.21
   continues in 005 seconds


(Caution: Not all Linux terminals can display the
 complete range of colors that GRUB supports)
```

The last configuration screen allows choosing a highlight color (foreground/background) for the choice display. Enter finishes the boot configuration. A final screen shows the output written into the configuration file **`grub-switch/bootfiles/.entries.txt`**, which is used to configure the switchable USB storage device.

```
The following data was written to ../bootfiles/.entries.txt:

### #1 specifies display time for subsequent entries
### #2 specifies display colors for subsequent entries
### Uncommented lines are GRUB switch choices 1..15 (0x1..0xF)
### An empty line leads to the GRUB menu, as does choice 0
#1 005
#2 white/red
Windows Boot Manager (on /dev/nvme0n1p1)
Ubuntu
Advanced options for Ubuntu>Ubuntu, with Linux 5.4.0-73-generic (recovery mode)
System setup
```

After pressing a key, the tool returns to the main screen. The status of generated bootfiles shows that they have just been generated.

```
Generated boot files for regular flash drives (../bootfiles/boot.[1..f]):
-> last updated at 25 Mai 2021 - 22:25:21
Generated boot file for GRUBswitch USB device (../bootfiles/.entries.txt):
-> last updated at 25 Mai 2021 - 22:25:21
```

To clean up, the generated files can be deleted with menu option **3**.

3.3 Storing hash values for boot choice authentication

For every generated boot file, a unique SHA512 hash sum been generated. By selecting action **5**, these hashes can be stored into the system's boot directory. As this option writes into the boot path, both *sudo* rights authorization and an extra confirmation are required:

```
5 - Install up-to-date hashes
-----------------------------
This action will install up-to-date hashes of permitted SWITCH.GRB files to /boot/grub/
Do you want to proceed? (y)es / (n)o
```

When such hashes have been installed, GRUB Switch will only execute boot options for which it finds corresponding hash sums. This mechanism protects the system against flash drives with unauthorized GRUB scripts.

```
Permitted SWITCH.GRB file hashes  (/boot/grub//grub_switch_hashes/*):
-> last updated at 25 Mai 2021 - 22:31:32
```

The hashes can be removed again from the boot directory with option **6**.

IMPORTANT NOTES:

1. *Even on Linux distributions that work with UEFI Secure Boot (only standard Ubuntu and openSUSE "Leap" in our tests), the GRUB hash module will NOT work with Secure Boot. We blame this on how GRUB loads the hash module. You'll have to decide whether to disable Secure Boot or to remove hash checks again with option **6**.*
2. *Hash checking should not be used at all with Fedora Linux, as using the hash function there fails but returns SUCCESS (implying ANY* **`SWITCH.GRB`** *file is OK'd).*
3. *The same is true of openSUSE if you install it with the default **btrfs** file system. Hash checking currently works if you install openSUSE on an **ext4** partition.*

4 Adding GRUB Switch to your GRUB installation

Before your generated boot choice files can have any impact, you need to install the actual *GRUB Switch* modification to GRUB. This is accomplished with action **7** in the tool:

```
ACTIONS:
--------

1 - Extract all menu entries from grub.cfg
2 - Configure GRUBswitch order and generate bootfiles and hashes
3 -   Remove generated files
4 - Write GRUBswitch bootfile to GRUBswitch USB device        (requires sudo)

5 - Install up-to-date hashes for permitted SWITCH.GRB files  (requires sudo)
6 -   Remove all hashes, no permission checking               (requires sudo)

7 - Install GRUBswitch into grub.cfg                          (requires sudo)
8 -   Remove GRUBswitch from grub.cfg                         (requires sudo)

q - Quit
```

This action also requires *sudo write* rights and an additional confirmation:

```
7 - Install GRUBswitch into grub.cfg
-------------------------------------
This action will re-generate the grub.cfg file by calling update-grub
Do you want to proceed? (y)es / (n)o
```

File status will show the installed *GRUB Switch* modification:

```
GRUB menu config file  (/boot/grub//grub.cfg):
-> last modified at 25 Mai 2021 - 22:35:25
   contains up-to-date GRUBswitch code
```

I*MPORTANT: GRUB Switch needs to be installed* ***ONLY ONCE****, not every time you change the boot choices and generate new files. If you have downloaded a newer version of* GRUB Switch *where the GRUB modifications have changed, the file status will indicate this and you can reinstall.*

GRUB Switch can be removed again with menu option **8**.

5 Problems with specific Linux distributions

5.1 Linux Mint

Linux Mint is based on Ubuntu and uses a somewhat problematic mechanism to generate its individualized boot entries. When **`update-grub`** is called, it actually generates a **`grub.cfg`** with menu entries named *Ubuntu*. Then, every time the PC shuts down or reboots,a script is called that substitutes every mention of *Ubuntu* with *Linux Mint*.

As a consequence, you should extract menu entries right after rebooting and without calling **`update-grub`** before (which is optional). This way, the extracted entries contain *Linux Mint* and will be recognized on the next boot.

5.2 Elementary OS

Elementary OS puts its **`grub.cfg`** into a different location; this location needs to be specified when starting the GRUB Switch configuration:

```
./CONFIGURE_GRUBswitch.sh  -g /boot/efi/EFI/ubuntu/grub/
```

(Note: To heighten confusion, the regular location /boot/grub/grub.cfg is also maintained and updated by Elementary OS, however it is not actually used when GRUB is booting)

5.3 Fedora Linux

Fedora uses entry titles that include the kernel version. Therefore, regularly update and re-extract boot choices to boot the most current kernel version with The GRUB Switch.

Lastly, checking hashes with GRUB Switch does not currently work in Fedora, and unfortunately returns a *success* value, implying the hash check recognized the file. Please do not use hash checking with Fedora!

5.4 openSUSE

openSUSE's hash checking also fails by returning *success* values everytime IF it is installed with the default **btrfs** file system. If you install openSUSE on an **ext4** partition instead, hash checking works.

6 Method A to use GRUB Switch: Using regular USB flash drives for boot choice

Picking and configuring boot entries with option **2** in the **CONFIGURE_GRUBswitch.sh** tool has generated directories and files like these in your directory **grub-switch/bootfiles** :

bootfiles/boot.1/SWITCH.GRB

bootfiles/boot.2/SWITCH.GRB

bootfiles/boot.2/SWITCH.GRB

You will find **boot.*** dirs according to the positions you assigned during configuration. There won't be a **boot.0** dir because that choice is reserved for the GRUB menu. Each directory holds a **SWITCH.GRB** file that sets the variable for the corresponding boot choice when called by GRUB. You can open it with a text editor, though I don't recommend editing it by hand.

You will now need as many USB flash drives as you have picked boot choices (excluding the GRUB menu).

- Make sure each flash drive is formatted with a FAT file system (FAT12/16/32, exFAT, VFAT), not with NTFS or any other file system.
  You do not need to re-format an existing drive, or even delete its files (although you might want to for reasons of cleanliness, data protection, privacy, etc.).
- Copy each **SWITCH.GRB** file to a separate flash drive and mark the drive with the corresponding number, OS name or whatever helps you tell them apart.
- Before powering on your computer or rebooting, plug in the flash drive representing your boot choice.
- If you want the GRUB menu to appear, make sure none of the drives are plugged in.
- If multiple of your drives are plugged in by accident, boot choice will depend on the ordering of USB connections in your BIOS, which you might not be able to tell easily.

All further chapters are not relevant if you stick with this method to make your boot choice.

7 Method B to use GRUB Switch: Using hardware switches and a programmable USB board for boot choice

If you want to make your boot choice through an electromechanical switch, you need to use a programmable USB board, flash it with our firmware and connect a suitable switch to it.

Our firmware is designed for boards with the Microchip/Atmel ATmega32U4 and ATmega16U4 chips, which can act as a USB device. All related files can be found in **`grub-switch/2_usb_device`**

7.1 Supported boards

Currently the project includes pre-built firmware images for the following products:

- PJRC Teensy 2.0
- Arduino Micro
- SparkFun Pro Micro (3V and 5V versions) and various clones
- Adafruit ItsyBitsy 32u4 (3V and 5V versions)
- DFRobot Beetle
- Our self-designed, non-commercial Custom Hardware PCB (see 5.4.6)

It is fairly simple to adapt the firmware to another board with the ATmega32U4/16U4 and build a firmware image. See the detailed documentation for instructions.

7.2 Programming the boards with GRUB Switch firmware

On the Linux command-line, go to the **`grub-switch/2_usb_device/prebuilt_images`** directory, where you can find the suitable image and programming scripts.

7.2.1 Teensy 2.0

Teensy 2.0 has a proprietary bootloader which works with its own GUI programming tool, Teensy Loader. You can find precise installation and download instructions here:

https://www.pjrc.com/teensy/loader_linux.html

The corresponding image is **`prebuilt_images/GS_TEENSY20.hex`**

7.2.2 Arduino-compatible boards

Arduino Micro, **ItsyBitsy**, **Pro Micro** and **Beetle** all use a bootloader that is compatible with Arduino IDE, however the IDE does not have an option for downloading any random *.hex file. We therefore require a command-line tool called **avrdude**, which is available in many Linux repositories.

- Under Ubuntu, for example, you can install **avrdude** with

  ```
  sudo apt-get install avrdude
  ```

- To simplify the use of **avrdude**, we have provided a script in **`prebuilt_images`**. After plugging in your board, it can be started with:

  ```
  sudo ./program_grubswitch_avrdude.sh CORRECT_FIRMWARE.hex
  ```

  (The use of *sudo* might not be required on your installation)

- The script will ask you to put the connected board into *bootloader mode:*
  - For the **Arduino Micro** and **ItsyBitsy** *bootloader mode* can be entered by pushing the board's reset button twice.
  - The **Pro Micro** does not have a reset button; you have to short the **GND** and **RST** pads twice in short order with a wire.
  - The **Beetle** has a group of 6 tiny pads on the bottom side below its USB socket:

    1 3 5
    # 0 0
    0 0 0
    2 4 6

    Pad number 1 is square, not round.
    Short the round pads number 5 and 6 twice in short order to put **Beetle** into bootloader mode

- After putting your board into *bootloader mode*, you have between 5 and 10 seconds to press a keyboard key for the programming script to continue; any longer, and the board leaves *bootloader mode* again.

After successful firmware programming, the board identifies as a FAT12 storage device. Depending on your Linux GUI, you might get a device notification.

NOTE: There are separate firmware images for the 3V and 5V versions of both ItsyBitsy and Pro Micro, as they need different oscillator configurations. Downloading the wrong firmware will not cause harm, the firmware will just not function. The board can be brought into bootloader mode *again, and reprogrammed with the correct firmware.*

7.2.3 THE GRUB SWITCH custom USB hardware

Our custom USB device design relies on the *DFU bootloader* which is factory-built into each ATmega32U4/16U4 chip. There is an open source command-line tool called **dfu-programmer** which needs to be installed for firmware flashing.

- Under Ubuntu Linux, it can be installed with:

  ```
  sudo apt-get install dfu-programmer
  ```

- To simplify the use of **dfu-programmer**, we have provided a script in **`prebuilt_images`**. After plugging in your board, it can be started with:

  ```
  sudo ./program_grubswitch_dfu.sh GS_CUSTOMHW_REVC_32U4.hex
  ```
  or
  ```
  sudo ./program_grubswitch_dfu.sh GS_CUSTOMHW_REVC_16U4.hex
  ```
  depending on the chip used on the board.
  (The use of sudo might not be required on your installation)

- Before or after starting the script, the chip needs to be brought into *bootloader mode* by pushing or shorting the **PROG** button. Unlike the Arduino boards, there is no bootloader timeout. Pressing a key on the keyboard will start programming.

After successful firmware programming, the board identifies as a FAT storage device with the drive name **BOOTTHIS**. Depending on your Linux GUI, you might get a notification.

7.3 Writing the GRUB Switch boot configuration onto your storage device

Together with the **`boot.*`** directories, option **`2`** of the **`CONFIGURE_GRUBswitch.sh`** script has also written a single text file holding the complete list and order of boot choices:

```
bootfiles/.entries.txt
```

You can open with a text editor to see the ordered list of boot choices, however it is not recommended to edit the file by hand. Because the filename starts with a period, it is only visible if you use the **`-a`** option with **`ls`**:

```
ls -la bootfiles/
```

If you mount your USB storage device automatically through the GUI, you can just copy the file onto the device and unmount again.

Alternatively, you can use option **`4 - Write GRUBswitch bootfile to GRUBswitch USB device`** in the **`CONFIGURE_GRUBswitch.sh`** tool. This requires giving the *sudo* password because the mounting and unmounting of the device is being forced.

NOTE: Your USB device is not a regular flash drive. **`.entries.txt`** is the only file that will be accepted for writing, and only if the **Write-Protect** jumper is not set on the board (see wiring section). The only other (read-only) files on the device are:

`SWITCH.GRB` – The actual boot setting script read by GRUB. If there are no switch connections on the board yet, this will be empty because that is the default leading to the GRUB menu.

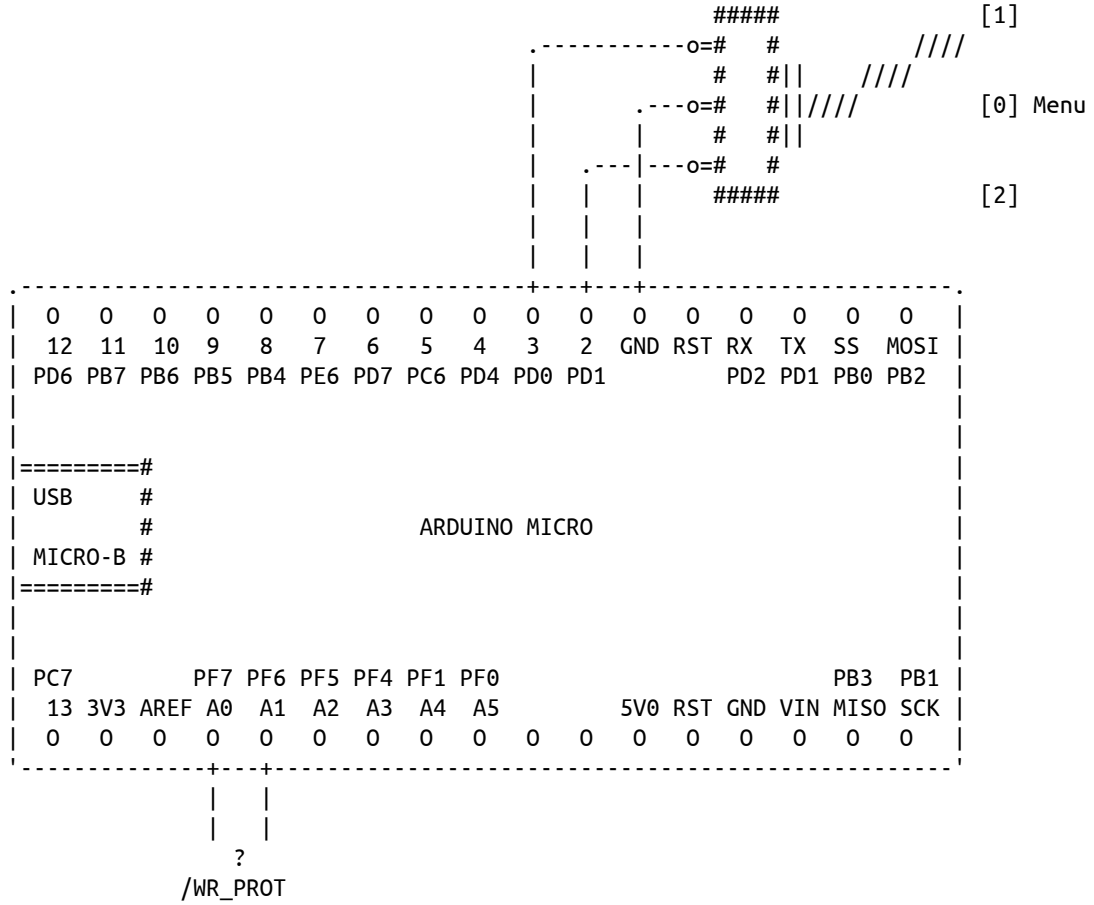
`.bootpins.txt` – This file (also hidden without **`ls -a`**) shows the choice pins currently connected to GND as ones. It is only included for testing purposes.

7.4 Electrical connections to the GRUB Switch boards

There are multiple ways to connect switches to each board to control the boot choice.

7.4.1 Arduino Micro

```
1-OF-N MODE  with  ON/OFF/ON toggle switch:
-------------------------------------------

To choose this mode, pin PF1 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Using a three-position on/off/on switch permits choosing between two
OS options, while the neutral middle position leads to the GRUB menu
```

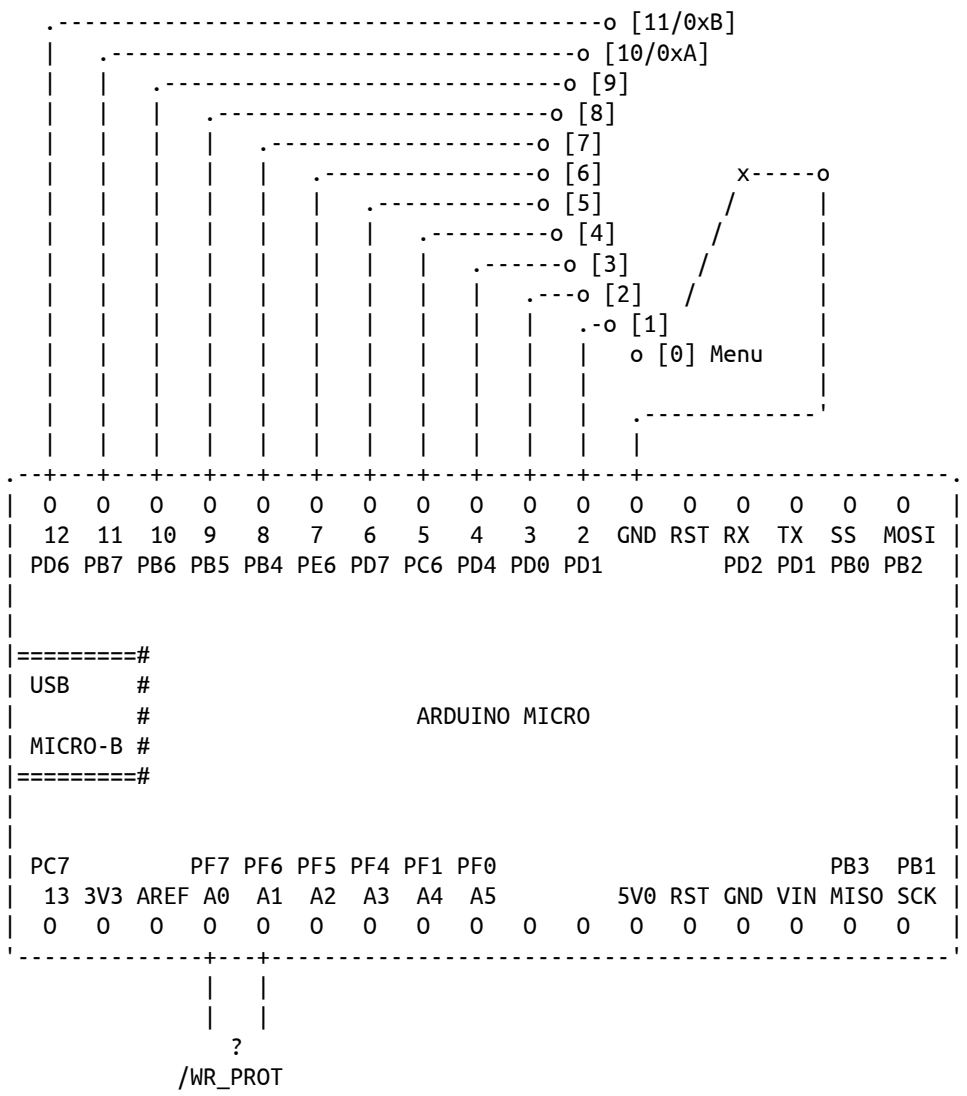


1-OF-N MODE with rotary switch: (Motto: "This one goes to 11!")

To choose this mode, pin PF1 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Common rotary switches have up to 12 positions (some can be limited to fewer
positions with a mechanical keyring). Keeping the first position [0] unconnected
permits choosing the GRUB menu and leaves 11 boot choices, which is the maximum
number of pins sampled. Connecting one of the pins PD1..PD6 to GND
therefore reflects boot choices 1..11 (hexadecimal 0x1..0xB). Accidentally
connecting multiple pins results in the lowest-order entry being chosen.

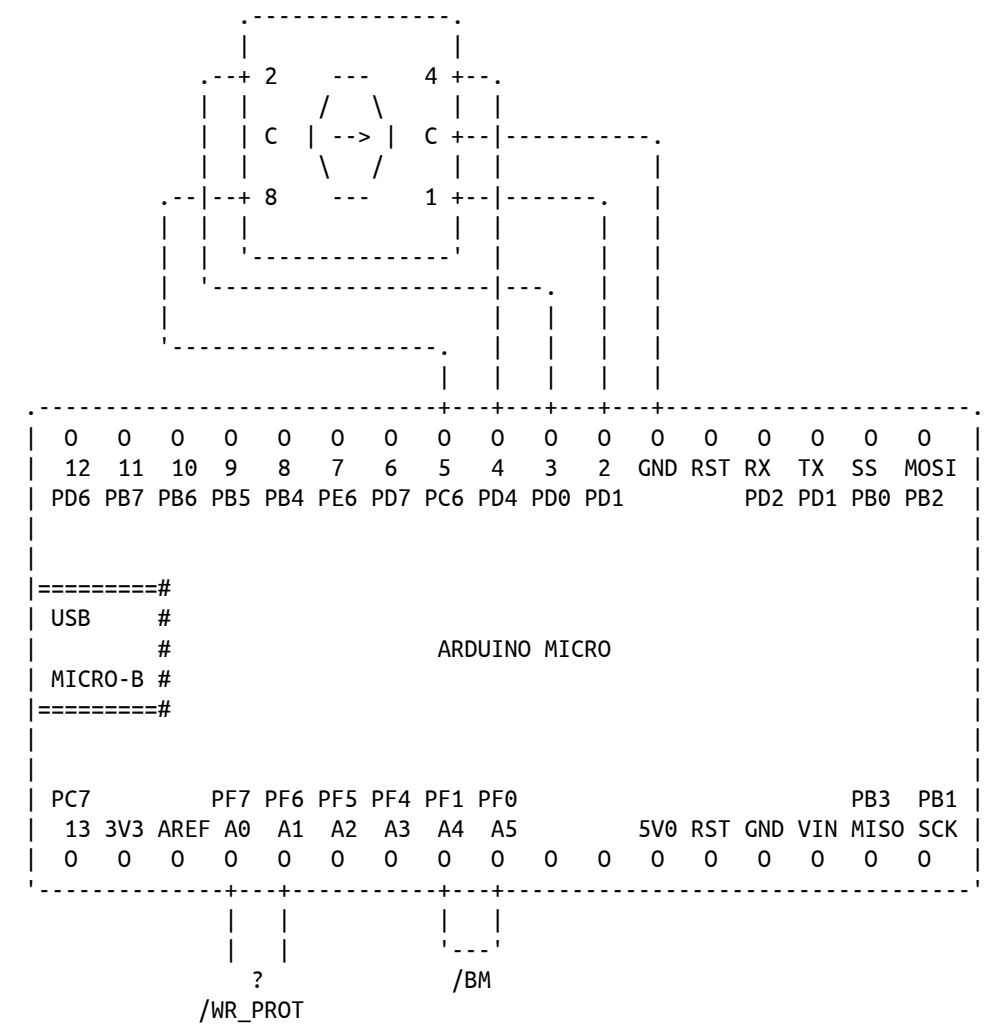


BINARY MODE:

To choose this mode, pin PF1 needs to be tied to GND or adjacent pad PF0.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Common binary-encoding rotary switches have 10 to 16 positions and connect
the corresponding bit values 1,2,4,8 [LSB..MSB] to the 'C'-contacts; using the
device's binary mode means tieing the four pins PD1..PC6 [LSB..MSB] to GND/0V
as to encode the corresponding boot choices 0(GRUB Menu) up to 15(0xF).
Obviously a proper nerd can also use four separate switches to encode the
choice in binary herself!
(Reminder: The device interprets a pin tied to GND as a '1', so all four
connections mean choice 15/0xF, while no connection means 0.



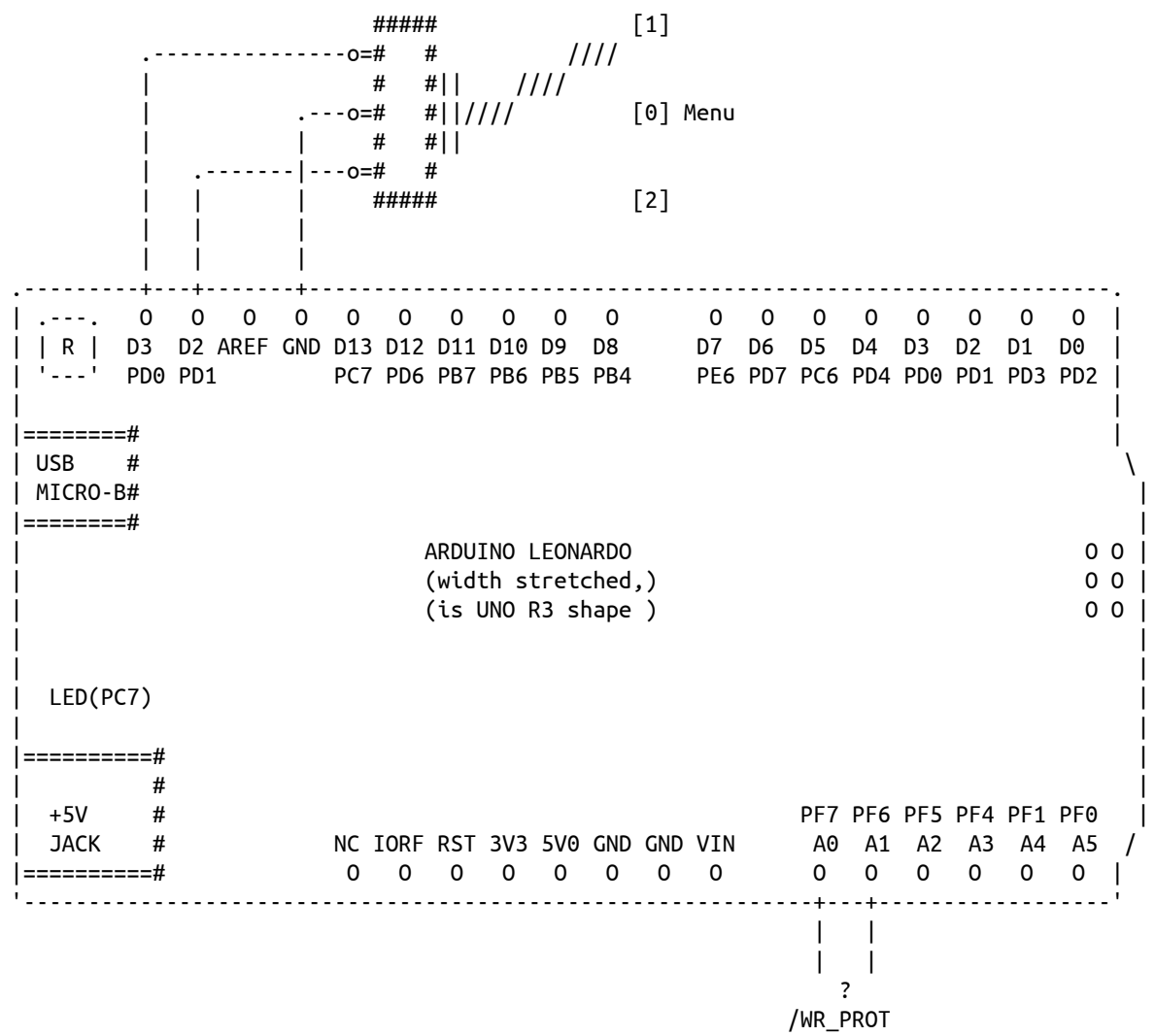
7.4.2 Arduino Leonardo

1-OF-N MODE with ON/OFF/ON toggle switch:

To choose this mode, pin PF1 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Using a three-position on/off/on switch permits choosing between two
OS options, while the neutral middle position leads to the GRUB menu

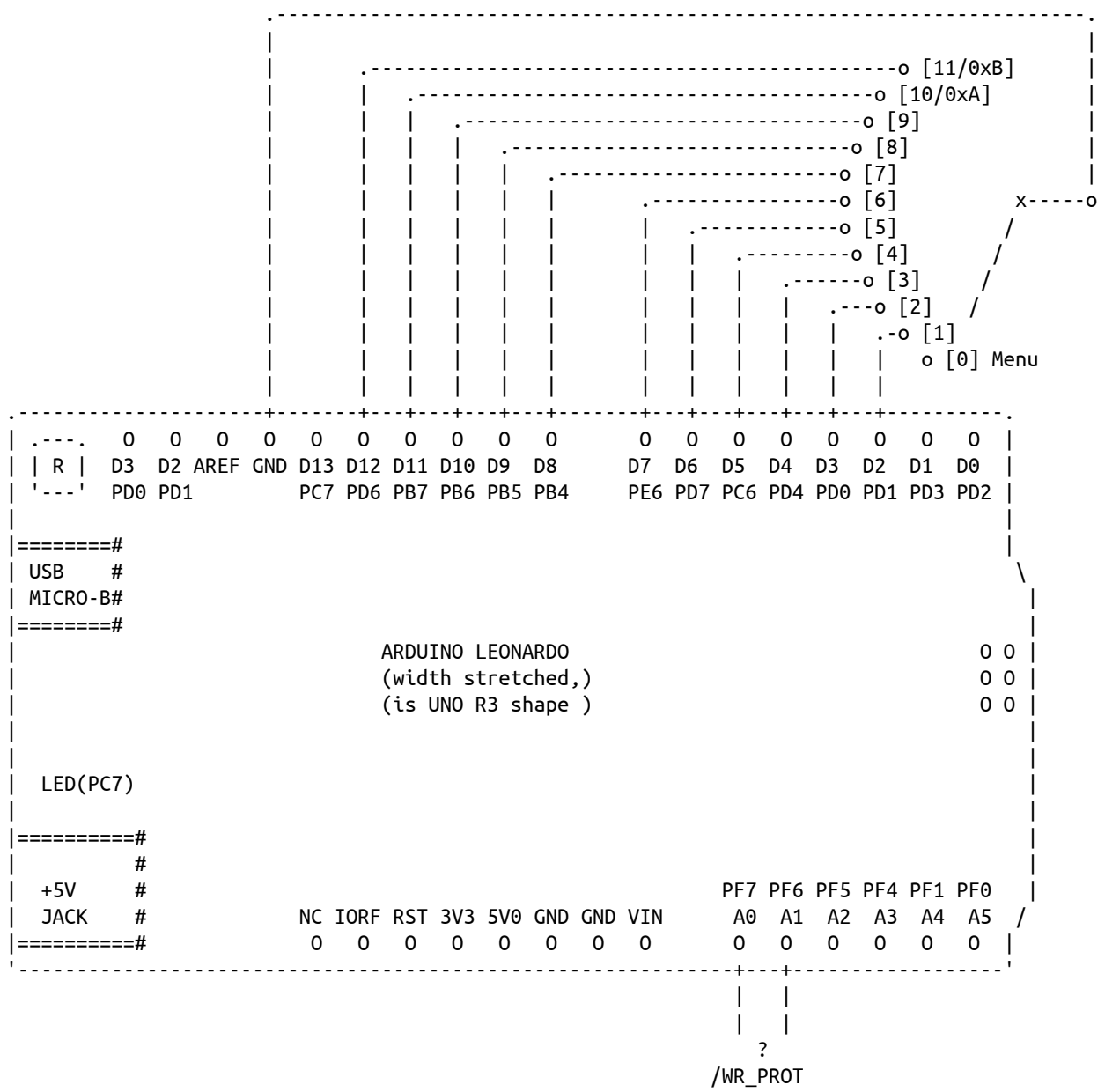


1-OF-N MODE with rotary switch: (Motto: "This one goes to 11!")

To choose this mode, pin PF1 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Common rotary switches have up to 12 positions (some can be limited to fewer
positions with a mechanical keyring). Keeping the first position [0] unconnected
permits choosing the GRUB menu and leaves 11 boot choices, which is the maximum
number of pins sampled. Connecting one of the pins PD1..PD6 to GND
therefore reflects boot choices 1..11 (hexadecimal 0x1..0xB). Accidentally
connecting multiple pins results in the lowest-order entry being chosen.

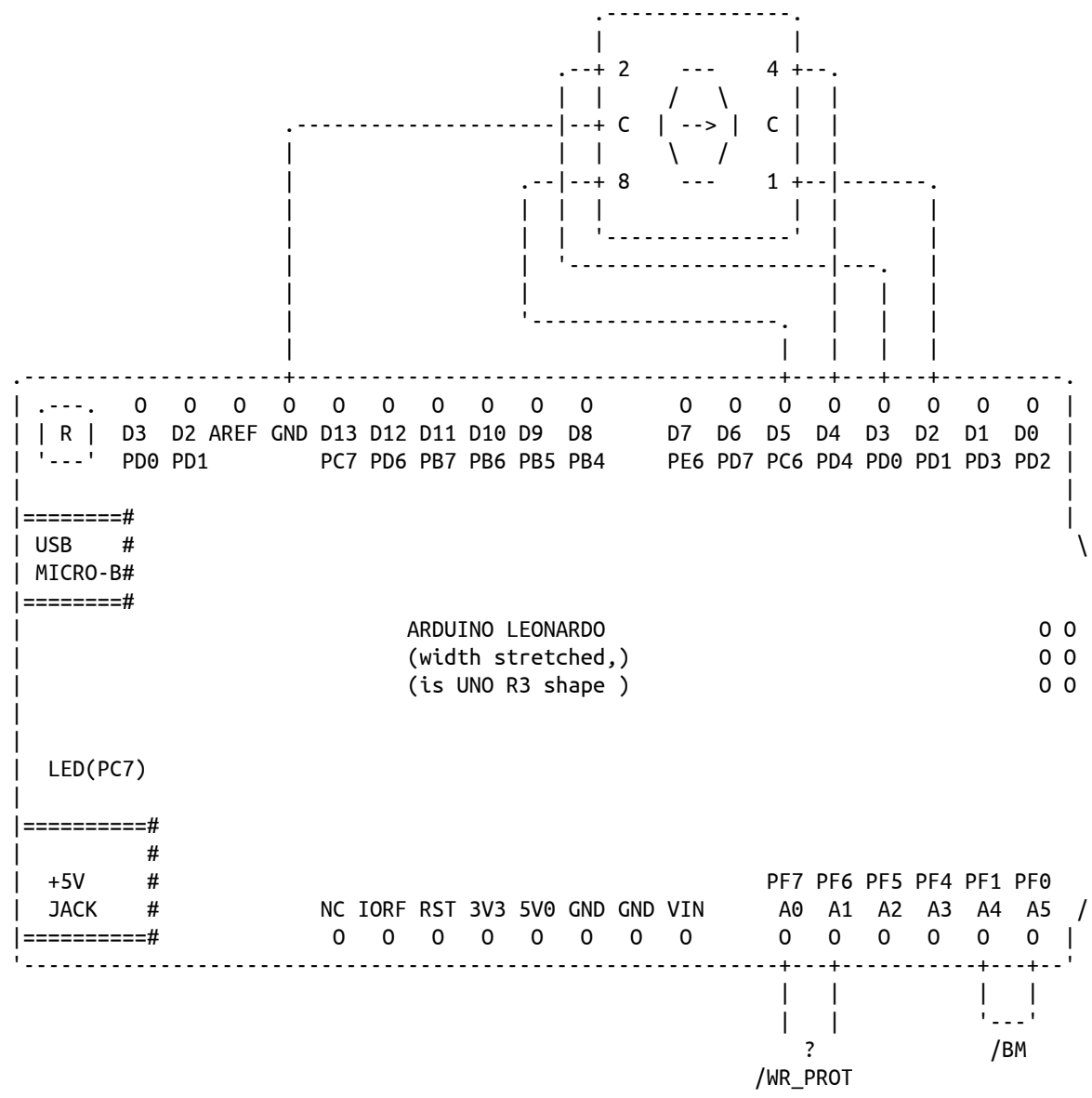


BINARY MODE:

To choose this mode, pin PF1 needs to be tied to GND or adjacent pad PF0.

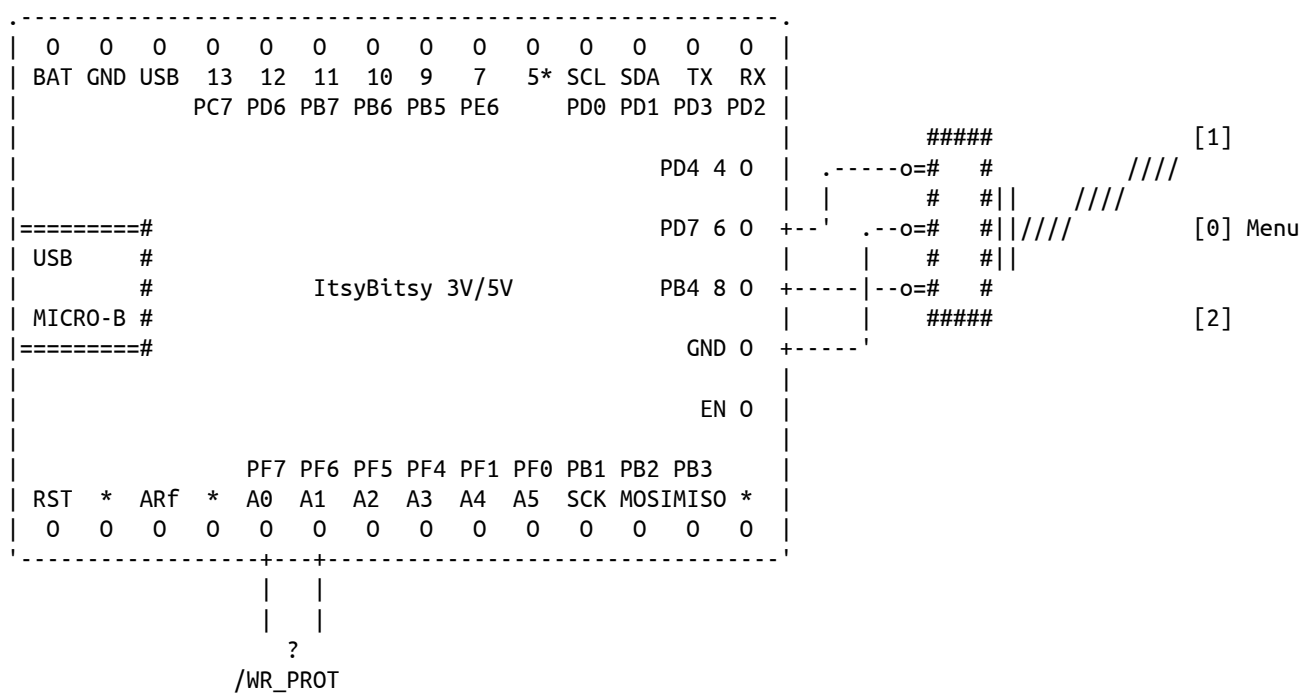
The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Common binary-encoding rotary switches have 10 to 16 positions and connect
the corresponding bit values 1,2,4,8 [LSB..MSB] to the 'C'-contacts; using the
device's binary mode means tieing the four pins PD1..PC6 [LSB..MSB] to GND/0V
as to encode the corresponding boot choices 0(GRUB Menu) up to 15(0xF).
Obviously a proper nerd can also use four separate switches to encode the
choice in binary herself!
(Reminder: The device interprets a pin tied to GND as a '1', so all four
connections mean choice 15/0xF, while no connection means 0.



7.4.3 Adafruit ItsyBitsy 3V/5V

```
1-OF-N MODE  with  ON/OFF/ON toggle switch:
-------------------------------------------

To choose this mode, pin PF1 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Using a three-position on/off/on switch permits choosing between two
OS options, while the neutral middle position leads to the GRUB menu
```

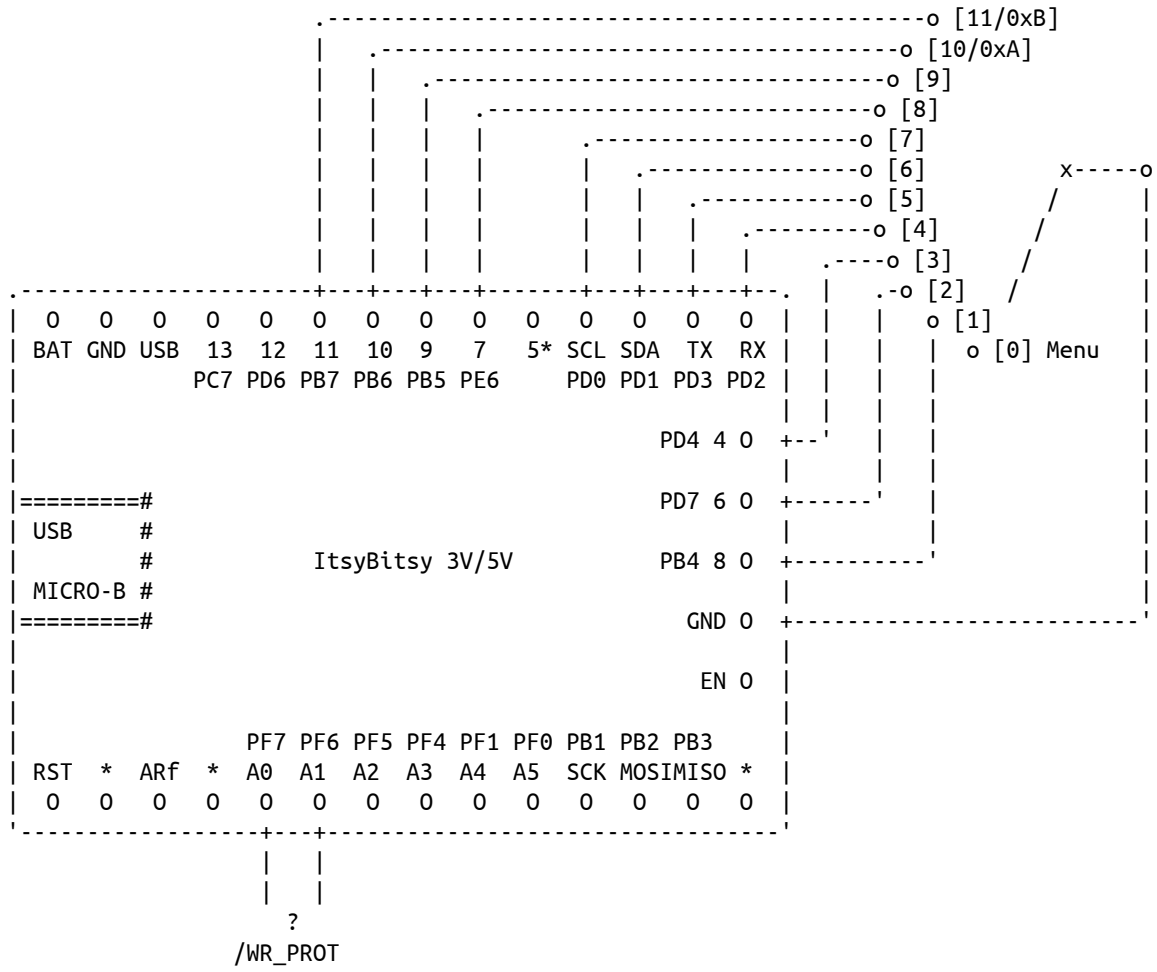


1-OF-N MODE with rotary switch: (Motto: "This one goes to 11!")

To choose this mode, pin PF1 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Common rotary switches have up to 12 positions (some can be limited to fewer
positions with a mechanical keyring). Keeping the first position [0] unconnected
permits choosing the GRUB menu and leaves 11 boot choices, which is the maximum
number of pins sampled. Connecting one of the pins PB4..PB7 to GND
therefore reflects boot choices 1..11 (hexadecimal 0x1..0xB). Accidentally
connecting multiple pins results in the lowest-order entry being chosen.

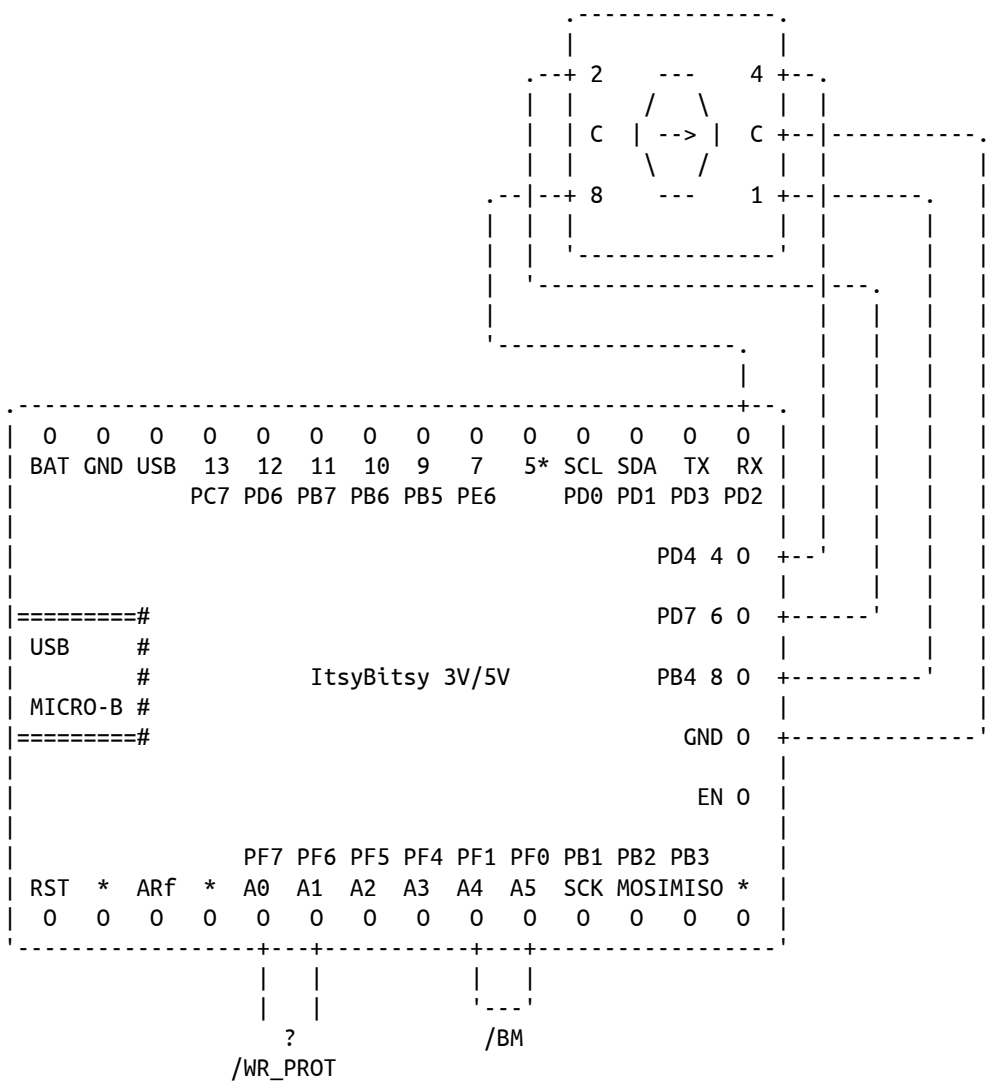


BINARY MODE:

To choose this mode, pin PF1 needs to be tied to GND or adjacent pad PF0.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF6 to GND or PF7 (adjacent pad, driving a low level).

Common binary-encoding rotary switches have 10 to 16 positions and connect
the corresponding bit values 1,2,4,8 [LSB..MSB] to the 'C'-contacts; using the
device's binary mode means tieing the four pins PB4..PD2 [LSB..MSB] to GND/0V
as to encode the corresponding boot choices 0(GRUB Menu) up to 15(0xF).
Obviously a proper nerd can also use four separate switches to encode the
choice in binary herself!
(Reminder: The device interprets a pin tied to GND as a '1', so all four
connections mean choice 15/0xF, while no connection means 0.



7.4.4 Sparkfun Pro Micro 3V/5V (and clones)

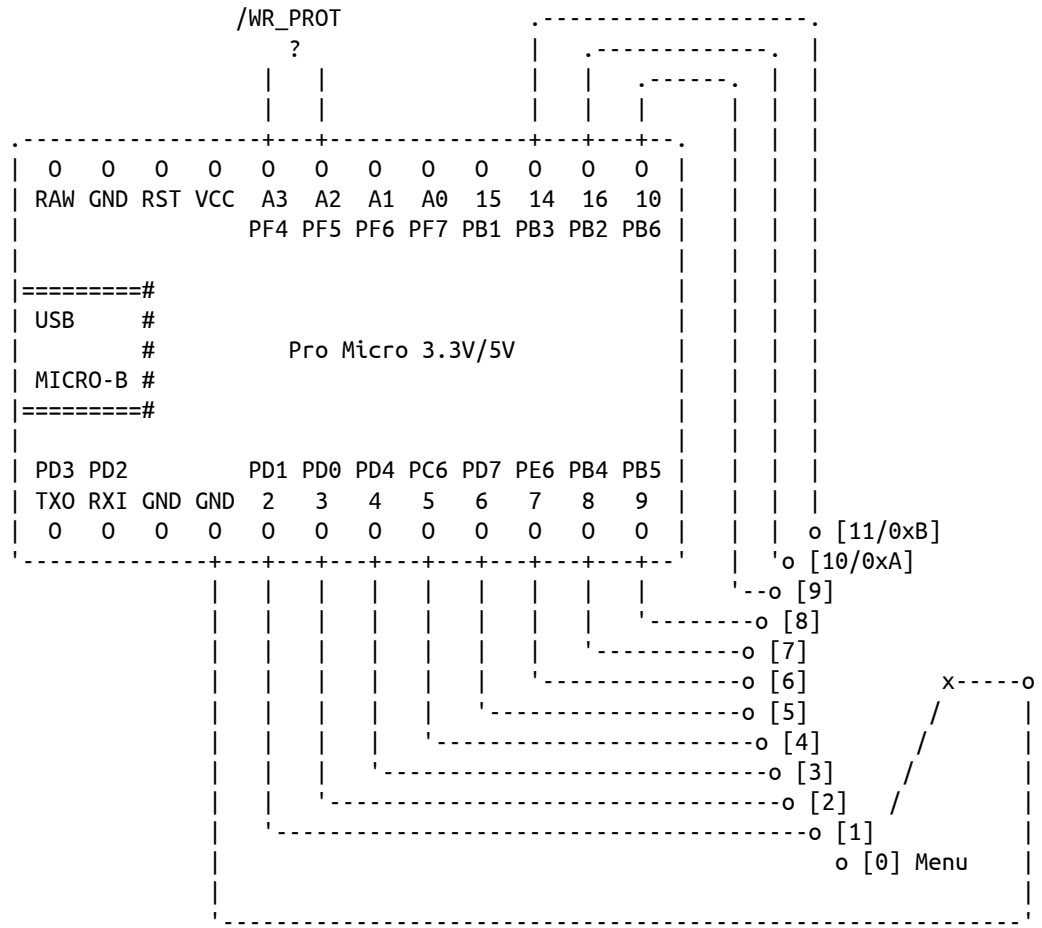
```
1-OF-N MODE  with  ON/OFF/ON toggle switch:
-------------------------------------------

To choose this mode, pin PD2 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF4 to GND or PF5 (adjacent pad, driving a low level).

Using a three-position on/off/on switch permits choosing between two
OS options, while the neutral middle position leads to the GRUB menu


                 /WR_PROT
                     ?
                   |   |
                   |   |
.------------------+---+--------------------------.
|  O   O   O   O   O   O   O   O   O   O   O   O  |
| RAW GND RST VCC  A3  A2  A1  A0  15  14  16  10 |
|                 PF4 PF5 PF6 PF7 PB1 PB3 PB2 PB6 |
|                                                 |
|=========#                                       |
| USB     #                                       |
|         #          Pro Micro 3.3V/5V            |
| MICRO-B #                                       |
|=========#                                       |
|                                                 |
| PD3 PD2         PD1 PD0 PD4 PC6 PD7 PE6 PB4 PB5 |
| TXO RXI GND GND  2   3   4   5   6   7   8   9  |
|  O   O   O   O   O   O   O   O   O   O   O   O  |
'--------------+---+---+--------------------------'
               |   |   |
               |   |   |             #####              [1]
               |   |   '-----------o=#   #          ////
               |   |                 #   #||    ////
               |   '---|-------------o=#   #|||||         [0] Menu
               |       |               #   #||
               |       '-------------o=#   #
                                       #####              [2]
```

1-OF-N MODE with rotary switch: (Motto: "This one goes to 11!")

To choose this mode, pin PD2 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF4 to GND or PF5 (adjacent pad, driving a low level).

Common rotary switches have up to 12 positions (some can be limited to fewer
positions with a mechanical keyring). Keeping the first position [0] unconnected
permits choosing the GRUB menu and leaves 11 boot choices, which is the maximum
number of pins sampled. Connecting one of the pins PD1..PB3 to GND
therefore reflects boot choices 1..11 (hexadecimal 0x1..0xB). Accidentally
connecting multiple pins results in the lowest-order entry being chosen.

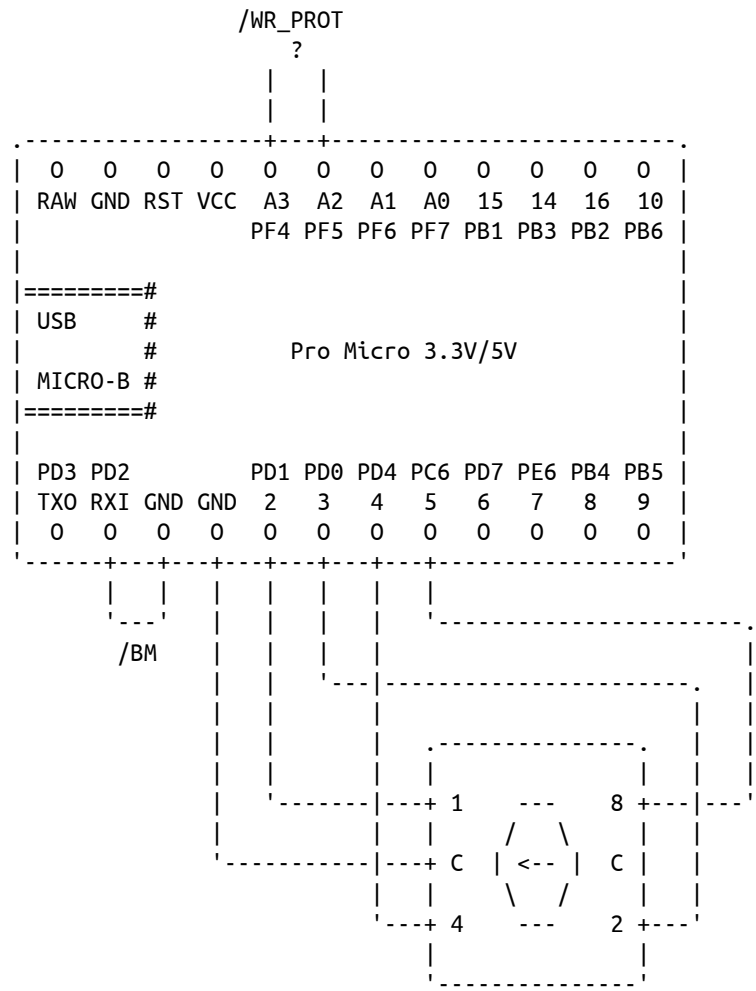


BINARY MODE:

To choose this mode, pin PD2 needs to be tied to the adjacent GND pin.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF4 to GND or PF5 (adjacent pad, driving a low level).

Common binary-encoding rotary switches have 10 to 16 positions and connect
the corresponding bit values 1,2,4,8 [LSB..MSB] to the 'C'-contacts; using the
device's binary mode means tieing the four pins PD1..PC6 [LSB..MSB] to GND/0V
as to encode the corresponding boot choices 0(GRUB Menu) up to 15(0xF).
Obviously a proper nerd can also use four separate switches to encode the
choice in binary herself!
(Reminder: The device interprets a pin tied to GND as a '1', so all four
connections mean choice 15/0xF, while no connection means 0.



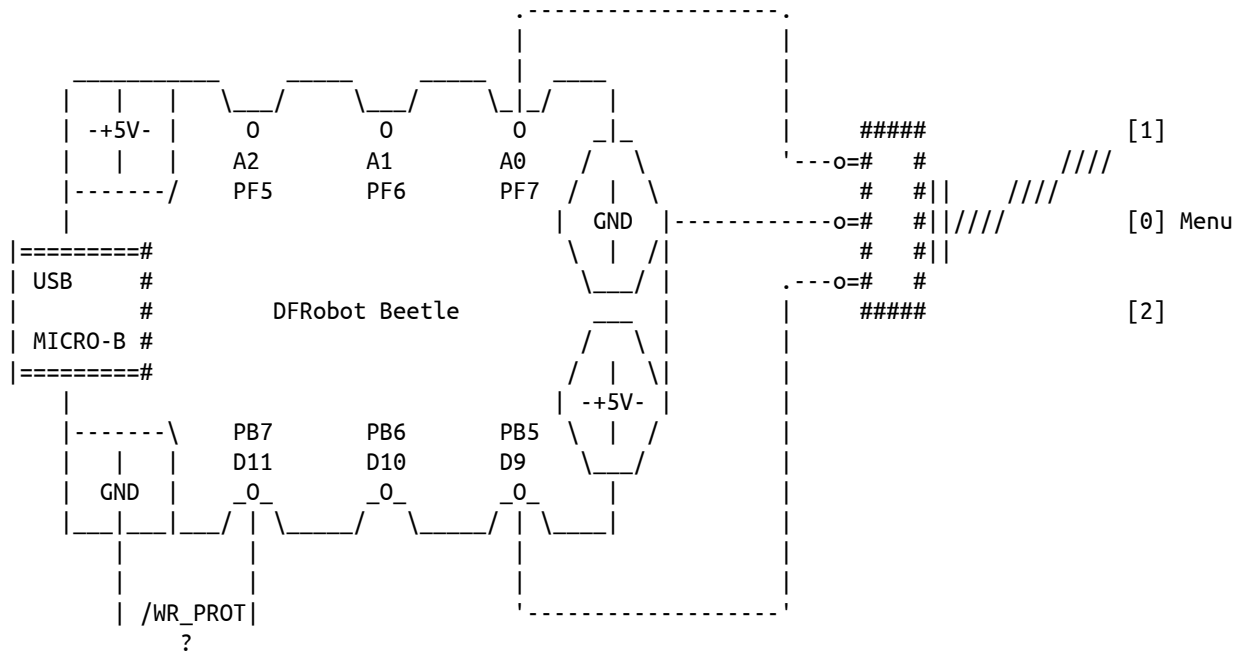
7.4.5 DFRobot Beetle

1-OF-N MODE with ON/OFF/ON toggle switch:

To choose this mode, pin PB6 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PB7 to GND.

Using a three-position on/off/on switch permits choosing between two
OS options, while the neutral middle position leads to the GRUB menu

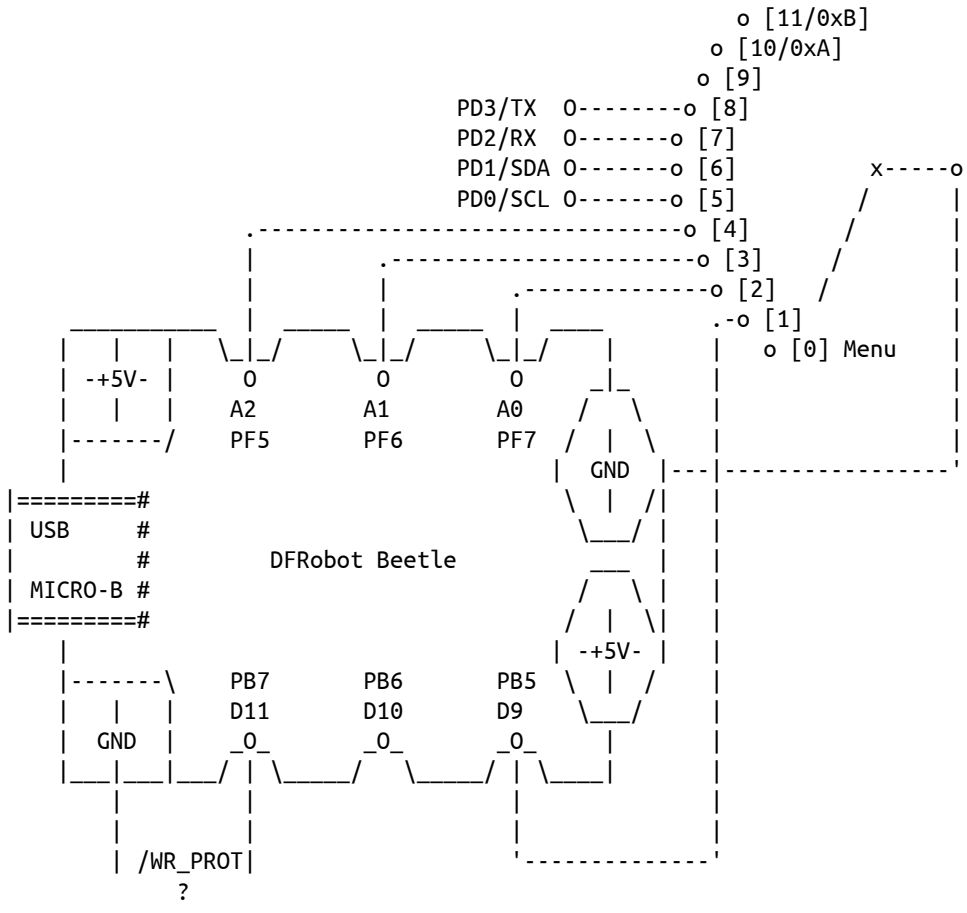


1-OF-N MODE with rotary switch: (Motto: "This one goes to 11!")

To choose this mode, pin PB6 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PB7 to GND.

Common rotary switches have up to 12 positions (some can be limited to fewer
positions with a mechanical keyring). Keeping the first position [0] unconnected
permits choosing the GRUB menu and in principle leaves 11 boot choices, however
Beetle only has 8 accessible connection points left, side pads PB5..BF5 for
positions [1]..[4] and the PCB bottom pads PD0..PD3 for positions [5]..[8]
Connecting one of those pads to GND through the switch reflects the boot
choices 1..8. Accidentally connecting multiple pins results in the lowest-order
entry being chosen.

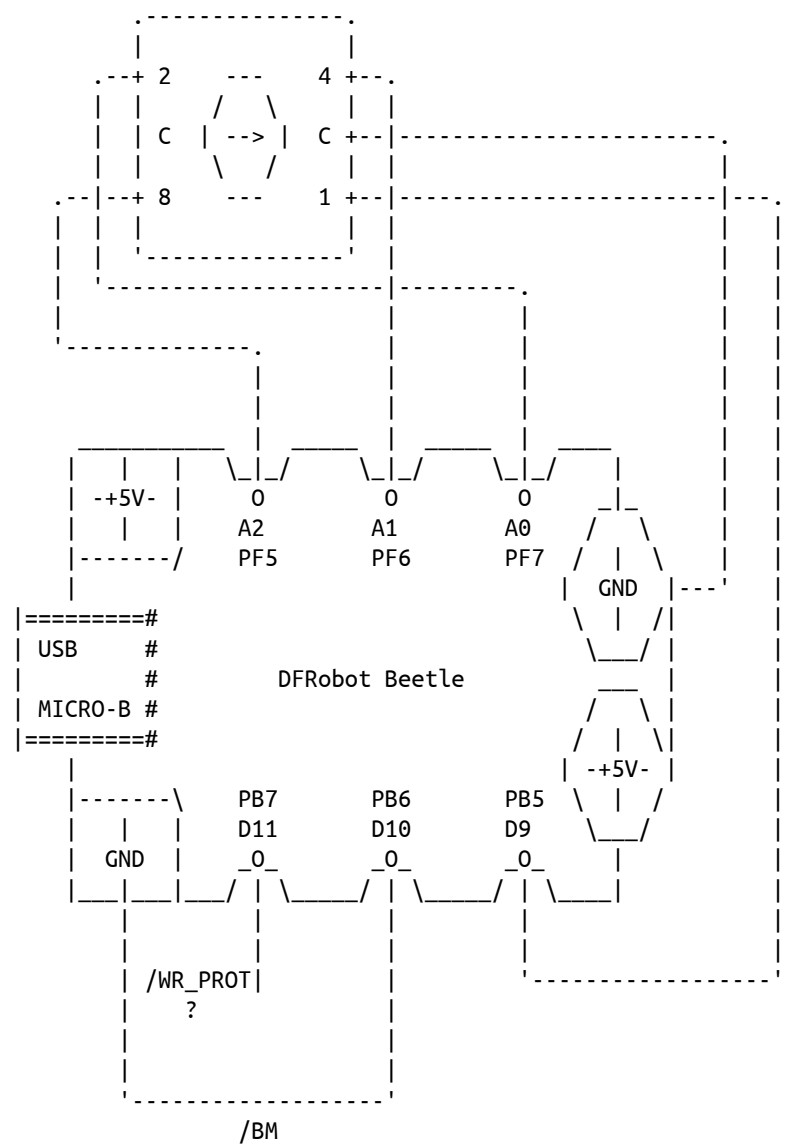


BINARY MODE:

To choose this mode, pin PB6 needs to be tied to GND.

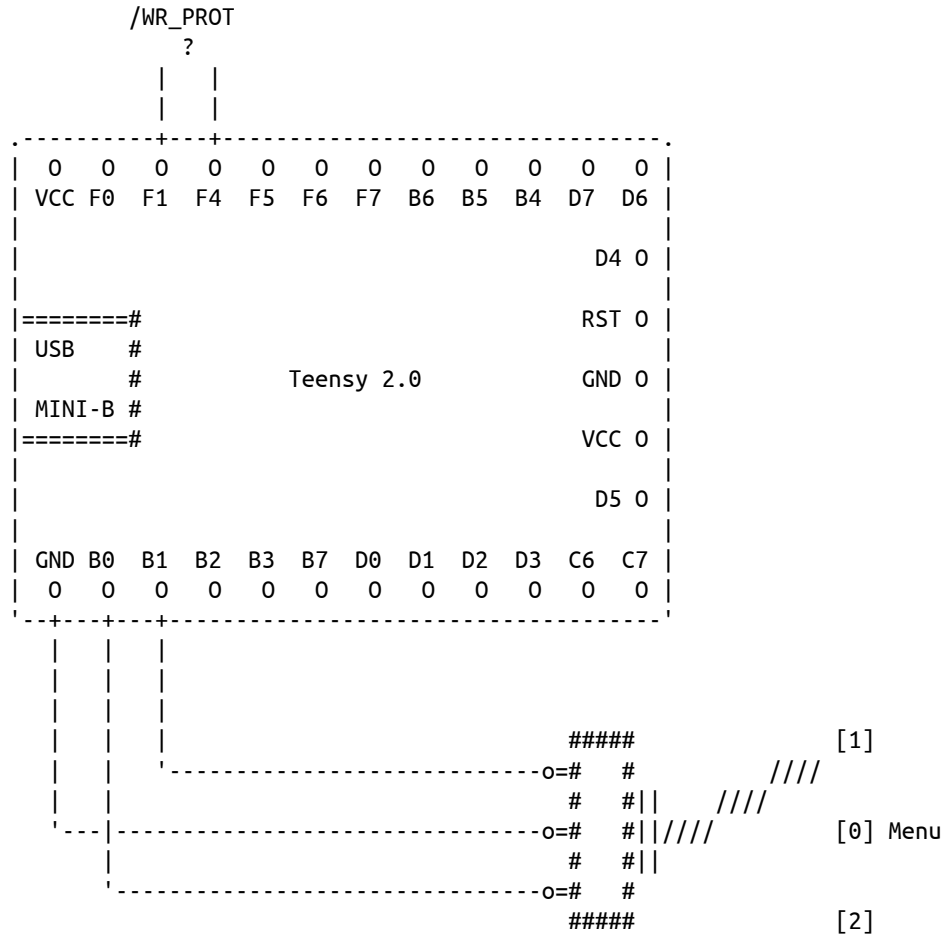
The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PB7 to GND.

Common binary-encoding rotary switches have 10 to 16 positions and connect
the corresponding bit values 1,2,4,8 [LSB..MSB] to the 'C'-contacts; using the
device's binary mode means tieing the four pins PB5..PF5 [LSB..MSB] to GND/0V
as to encode the corresponding boot choices 0(GRUB Menu) up to 15(0xF).
Obviously a proper nerd can also use four separate switches to encode the
choice in binary herself!
(Reminder: The device interprets a pin tied to GND as a '1', so all four
connections mean choice 15/0xF, while no connection means 0.



7.4.6 PJRC Teensy 2.0

```
1-OF-N MODE  with  ON/OFF/ON toggle switch:
-------------------------------------------

To choose this mode, pin PB6 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF4 to GND or PF1 (adjacent pad, driving a low level).

Using a three-position on/off/on switch permits choosing between two
OS options, while the neutral middle position leads to the GRUB menu
```

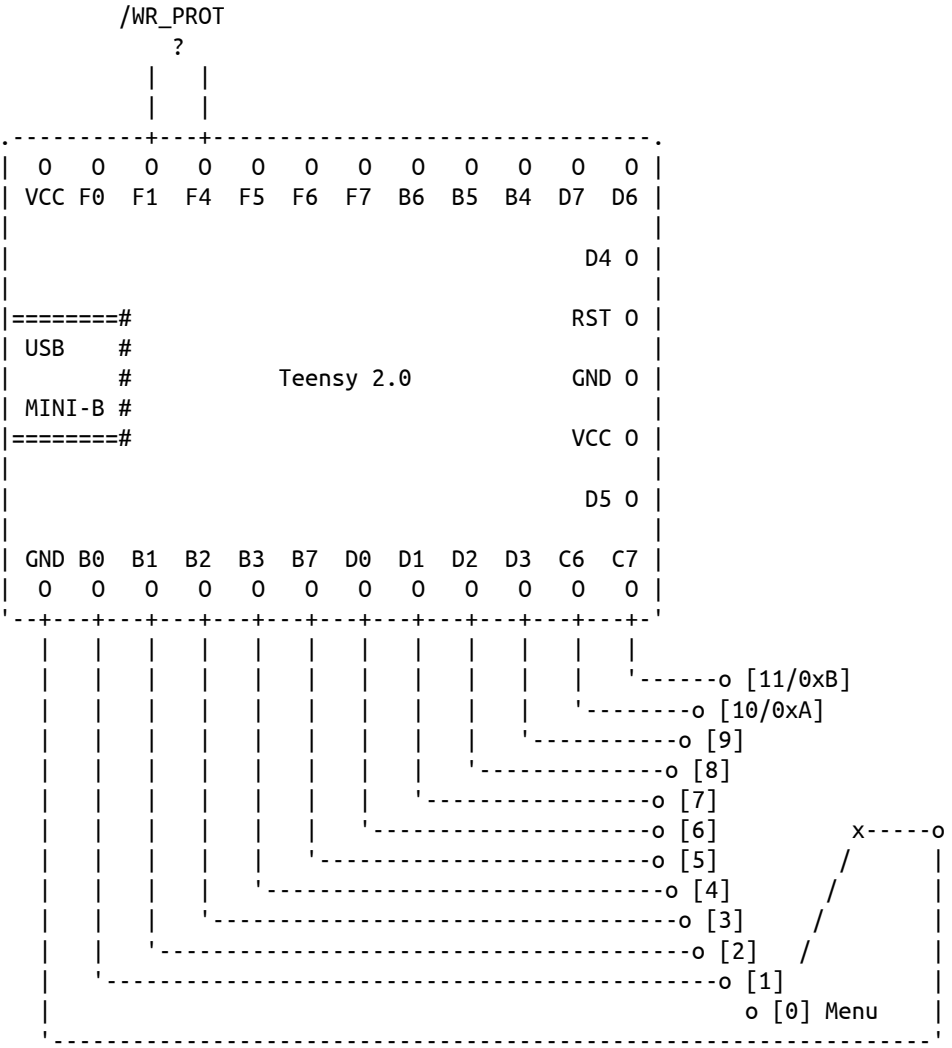


1-OF-N MODE with rotary switch: (Motto: "This one goes to 11!")

To choose this mode, pin PB6 needs to stay UNCONNECTED.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF4 to GND or PF1 (adjacent pad, driving a low level).

Common rotary switches have up to 12 positions (some can be limited to fewer
positions with a mechanical keyring). Keeping the first position [0] unconnected
permits choosing the GRUB menu and leaves 11 boot choices, which is the maximum
number of pins sampled. Connecting one of the pins PB0..PC7 to GND
therefore reflects boot choices 1..11 (hexadecimal 0x1..0xB). Accidentally
connecting multiple pins results in the lowest-order entry being chosen.

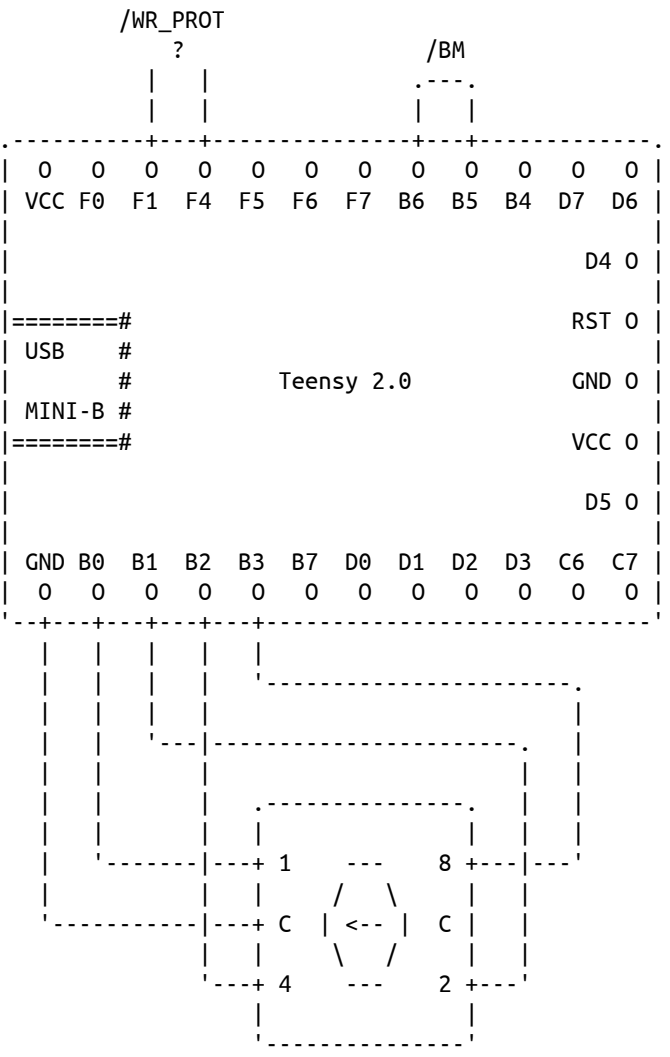


BINARY MODE:

To choose this mode, pin PB6 needs to be tied to GND or adjacent pad PB5.

The Write Protect against modifying boot entry configurations is activated
by tieing/jumpering pin PF4 to GND or PF1 (adjacent pad, driving a low level).

Common binary-encoding rotary switches have 10 to 16 positions and connect
the corresponding bit values 1,2,4,8 [LSB..MSB] to the 'C'-contacts; using the
device's binary mode means tieing the four pins PB0..PB3 [LSB..MSB] to GND/0V
as to encode the corresponding boot choices 0(GRUB Menu) up to 15(0xF).
Obviously a proper nerd can also use four separate switches to encode the
choice in binary herself!
(Reminder: The device interprets a pin tied to GND as a '1', so all four
connections mean choice 15/0xF, while no connection means 0.



7.4.7 The GRUB SWITCH Custom Hardware

For the switch connection options available for our custom hardware PCB, please refer to its full documentation in

`grub-switch/documentation/3_custom_hardware.pdf`