1. Developer's Guide

This includes instructions on how to burn a new firmware to your SOAR.

2. Table of Contents

3. Flashing a pre-built `.hex` file

3.1. Install `TeensyLoader` (Do this once)

Download and install the Teensy Loader for your OS of choice: https://www.pjrc.com/teensy/loader.html

NOTE FOR LINUX USERS Make sure you follow all the instructions on that page, especially the part about putting a file in `/etc/udev/rules.d/`. Without this, your user account won't have permissions to talk to the USB sub-system.

Plug in your SOAR to a USB port on your computer, and start the `TeensyLoader`. (It doesn't matter what order this is done in.) If `TeensyLoader` can see the SOAR, the two arrow buttons (the "swooping down" arrow for `Program`, and the "right pointing arrow" for `Reboot`) will be green. If it can't see the SOAR, then they will be gray'd out.

If the SOAR is totally bricked (so that the boot loader isn't even running), the arrows will be gray'd out too. That (probably) ok, we can fix this.

3.2. Downloading New Firmware

New releases of firmware are available here: https://electronics.halibut.com/releases/soar/

Files are date stamped. Download the latest release. This will give you a file that ends with `.hex`. This is the firmware file. Note this filename and location.

3.3. Flash the Firmware

Click on the Document icon, `Open HEX File`. Load the `.hex` file you just downloaded.

- **If the `Program` button works**, then press it.
- **If the `Program` button is gray'd out or does not work**, then use a long plastic tool of some kind and press the (physical) `Program` button on the Microcontroller in the middle of the middle board.
    - "Golly, Mark, could you make that any harder to get to?" Sorry about that... I don't have to do this when programming from PlatformIO, it just programs the Teensy directly and automatically. I haven't yet figured out what PlatformIO is doing that makes this work automatically. I'll update this document if I figure it out. (I also accept pull requests! 😉

3.4. Getting At The SD Card (Version v2.1 only)

Note: This section only applies to the v2.1 Prototype hardware. This will probably be replaced when v2.2 Production hardware is available.

Note: This is (hopefully) no longer necessary, since the 2022-02-01 build, which adds MTP access to the SD card over USB.

There's an SD Card on the buried end of the Teensy. This stores all configuration, audio recordings, etc. It is presented to your computer over USB as an MTP device (like a camera or phone). If you can't get MTP working, or otherwise want direct access to this SD card, you'll need to remove the UI board on top to access the SD card:

- Remove all power (including USB cable) from SOAR
- Remove the four screws holing the UI Deck (the top board) from the stack.
    - Note: The UI Deck is the green board, not just the display on the red board. The display is part of the UI Deck, and is hard soldered down. Don't try to remove it.
- Carefully remove the UI Deck board from the stack. Note that there are three different headers connecting the UI Deck to the DA Deck (the middle board). Be careful not to bend these pins.
- **ONLY REMOVE OR INSERT THE SD CARD WHEN THE MICROCONTROLLER IS NOT POWERED**

The physical `Program` Button is also much easier to access with the UI Deck removed. It is safe to flash SOAR with the UI Deck removed.

Be very careful when replacing the UI Deck! The tall header pins between the UI Deck and DA Deck are very bendy. It may take a bit of "Insert the long strip of pins first, just a little bit, then kinda force the UI Deck to the side a bit while getting the middle pins to go in, then forcing it a different way to get the last three pins in" or something like this. It's a little fiddly.

4. Setting up a Full Development Environment

TODO

Quick notes:

- Install vsCode.

    - Install Add-ons: C++, Python (probably not necessary, but I did), Gitlense (also probablty not necessary)
- Install PlatformIO.
- Get the `SOAR-Code` source code from GitHub (currently (2022-04-05) a private repo, but this might change.)
- Open the `SOAR-Code/SOAR-v2-Code` directory in VS Code. This will download and install all the other addons that PlatformIO needs, like the Arduino platform, Teensy boards, and the libraries.
- Modify the Teensy Platform for `USB_SOAR` support:

    - Edit `.platformio/packages/framework-arduinoteensy/cores/teensy4/usb_desc.h` and add:

```
// Equivalent to MTPDISK_SERIAL_AUDIO, but with the manufacturer and
product names
// set to Halibut Electronics stuff.
#elif defined(USB_SOAR)
#define VENDOR_ID             0x16C0
#define PRODUCT_ID                0x04D5 //fake an include
everything device
#define RAWHID_USAGE_PAGE 0xFFAB  // recommended: 0xFF00 to 0xFFFF
#define RAWHID_USAGE      0x0200  // recommended: 0x0100 to 0xFFFF
#define DEVICE_CLASS      0xEF
#define DEVICE_SUBCLASS   0x02
#define DEVICE_PROTOCOL   0x01

#define MANUFACTURER_NAME     {'H','a','l','i','b','u','t','
','E','l','e','c','t','r','o','n','i','c','s'}
#define MANUFACTURER_NAME_LEN 19
#define PRODUCT_NAME          {'S','O','A','R','-','1'}
#define PRODUCT_NAME_LEN      6
#define EP0_SIZE              64
#define NUM_INTERFACE         6
#define NUM_ENDPOINTS         7

#define CDC_IAD_DESCRIPTOR    1 // Serial, last interface: 2, last
endpoint: 3
#define CDC_STATUS_INTERFACE  1
#define CDC_DATA_INTERFACE    2
#define CDC_ACM_ENDPOINT      2
#define CDC_RX_ENDPOINT       3
#define CDC_TX_ENDPOINT       3
#define CDC_ACM_SIZE          16
#define CDC_RX_SIZE_480       512
#define CDC_TX_SIZE_480       512
#define CDC_RX_SIZE_12        64
#define CDC_TX_SIZE_12        64
#define ENDPOINT2_CONFIG  ENDPOINT_RECEIVE_UNUSED +
ENDPOINT_TRANSMIT_INTERRUPT
```

```
#define ENDPOINT3_CONFIG  ENDPOINT_RECEIVE_BULK +
ENDPOINT_TRANSMIT_BULK

#define MTP_INTERFACE         3 // MTP Disk, last interface: 3, last
endpoint: 5
#define MTP_TX_ENDPOINT       4
#define MTP_RX_ENDPOINT       4
#define MTP_EVENT_ENDPOINT    5
#define MTP_TX_SIZE_480       512
#define MTP_RX_SIZE_480       512
#define MTP_TX_SIZE_12        64
#define MTP_RX_SIZE_12        64
#define MTP_EVENT_SIZE        32
#define MTP_EVENT_INTERVAL_12 10  // 10 = 10 ms
#define MTP_EVENT_INTERVAL_480 7  // 7 = 8 ms
#define ENDPOINT4_CONFIG  ENDPOINT_RECEIVE_BULK +
ENDPOINT_TRANSMIT_BULK
#define ENDPOINT5_CONFIG  ENDPOINT_RECEIVE_INTERRUPT +
ENDPOINT_TRANSMIT_INTERRUPT  // ????

#define AUDIO_INTERFACE       4   // Audio (uses 3 consecutive
interfaces), last interface: 6, last endpoint: 7
#define AUDIO_TX_ENDPOINT     6
#define AUDIO_TX_SIZE         180
#define AUDIO_RX_ENDPOINT     6
#define AUDIO_RX_SIZE         180
#define AUDIO_SYNC_ENDPOINT   7
#define ENDPOINT6_CONFIG  ENDPOINT_RECEIVE_ISOCHRONOUS +
ENDPOINT_TRANSMIT_ISOCHRONOUS
#define ENDPOINT7_CONFIG  ENDPOINT_RECEIVE_UNUSED +
ENDPOINT_TRANSMIT_ISOCHRONOUS
```

- Edit `.platformio/packages/framework-arduinoteensy/cores/teensy4/usb.c`, add `USB_SOAR` to the OR list on line 1035:

```
uint32_t usb_transfer_status(const transfer_t *transfer)
{
#if defined(USB_MTPDISK) || defined(USB_MTPDISK_SERIAL) ||
defined(USB_SOAR)
    uint32_t status, cmd;
[...]
```

- Edit `.platformio/platforms/teensy/builder/frameworks/arduino.py`, add `USB_SOAR` to the list of `BUILTIN_USB_FLAGS`.

- If only doing `USB_MT_SERIAL`, do these steps. (These aren't strictly necessary if using USB_SOAR, but don't conflict either.)
    - TBD: Need to get this off my XPS15. Using `USB_MTPDISK_SERIAL` in the mean time. Bonus: It doesn't hijack my music every time I reprogram the radio.
    - Modify `.platformio/packages/framework-arduinoteensy/cores/teensy4/` to add `USB_MTP_SERIAL` section
    - Modify `.platformio/platforms/teensy/builder/frameworks/arduino.py` to add `USB_MTP_SERIAL` to the list.
- Build, Upload, Profit!