Procedure: Build .elf File for Atmel with Arduino-based USB DFU Bootloader

1. GOALS AND RATIONALE

The aim is to build an .elf file for a board with an Atmel microcontroller. The .elf file combines fuse settings, an Arduino USB bootloader and the main program.

The production .elf file is built to simplify programming of boards at the contract manufacturer. The USB bootloader is used for ease of updating boards sealed in an enclosure, and to make field programming possible without a dedicated programming pod. Though the Atmel chip may have its own USB DFU bootloader, it really only works the first time (not for reprogramming), and the Atmel FLIP program is buggy and not really supported.

Knowledge of how to upload a program to an Arduino board with the IDE is assumed. Only Arduino will work with this process. Atmel code can be ported over with small changes: Arduino is looking for setup() and loop() functions in the main program file; otherwise all Atmel syntax, register addressing, etc., work in Arduino.

2. EQUIPMENT AND SOFTWARE

- Computer running Windows
  - Arduino IDE 1.8.5 software
  - Atmel Studio 7.0 software
  - Main firmware for target device, written as Arduino .ino file
- Modified Arduino Uno running Nick Gammon's "Atmega bootloader programmer"
- Tag-Connect TC2030-IDC-NL connector cable
- Atmel JTAGICE3 programming pod
- Tag-Connect TC2030-ICESPI-LEMTA connector cable
- Tag-Connect TC2030-CLIP retaining clip

Notes:
The Uno is modified to use the Tag-Connect cable. You can also just use jumpers if you don't want to mod the board. The mod will not allow the Uno to be programmed via ISP until modded back. The mod is simply to cut the trace from pin 1 of the Atmega328 (DIP version) to pin 5 of the nearby ISP connector. Then jumper pin 5 of the ISP to pin 16 of the Atmega328.
Nick Gammon's sketch can be found here: http://www.gammon.com.au/bootloader
It is used instead of ArduinoISP because it is much faster (a few seconds instead of over a minute).

3. LOADING BOOTLOADER

3.1. Connect Uno to computer and open IDE with Gammon’s sketch.

3.2. Load sketch into Uno if it’s not already in there.

3.3. Open serial monitor at 115200 baud.

3.4. Connect target board to Uno with Tag-Connect cable. Uno header is not keyed. Red strip is on the side closer to the digital I/O header.

3.5. Press the Reset button on Uno.

3.6. The serial monitor should announce itself ready to program if you type ‘G’. Do that.

3.7. In three seconds, the bootloader should be done. The board is effectively a blank Arduino now.

4. LOADING FIRMWARE

4.1. Connect the target board to the computer via USB.

4.2. It should pop up in Device Manager as the appropriate Arduino board (i.e., Leonardo for an Atmega32U4).

4.3. Open the main program sketch and upload like you would any Arduino program.

5. MAKING THE ELF

5.1. Connect JTAGICE pod to computer with USB cable.

5.2. Open Atmel Studio 7.0.

5.3. In ‘Tools’ menu, choose ‘Device Programming’.

5.4. Tool should be “JTAGICE3”, Device “ATmega32U4” (for example) and Interface “ISP”. Click ‘Apply’.

5.5. Connect JTAGICE pod to target board with Tag-Connect cable.

5.6. On the left side, click ‘Fuses’, then ‘Read’ on the fuses page.

5.7. On the left side, click ‘Memories’, then ‘Read’ on the memories page. Save .hex file to computer.

5.8. On the left side, click ‘Production file’.

5.9. In ‘Save to ELF production file’ section, choose Flash (.hex file from step 5.7), then click the boxes for Flash and Fuses.

5.10. Click ‘Save’, and that is the new production file that will load fuse settings, bootloader and main program in the target board.