Bootloader for dsPIC33F Devices

INTRODUCTION

The bootloader for dsPIC30F/33F devices is used to load and run your application on the target device. The bootloader consists of two applications:

- Target side bootloader application which must be programmed into dsPIC33F program memory prior to bootloader operation.
- Host PC bootloader application which communicates with the target side bootloader.

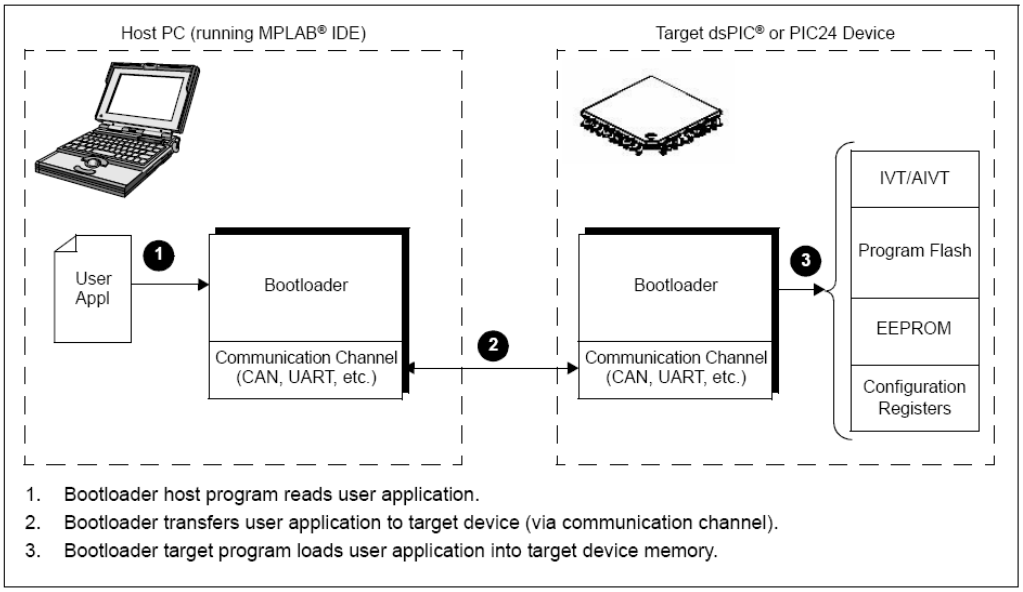
The bootloader parses the program HEX file and then copies it into the appropriate program memory on the target device via the communication channel (UART, CAN, etc.). Figure 1 illustrates this process.

SYSTEM CONCEPT

The bootloader target application is located in the dedicated program memory region, starting at address 0x400. On start-up, the bootloader reads program memory address 0xC00, which contains a bootloader delay value. If the bootloader fails to detect UART activity within the time period specified by the delay value, it suspends itself and transfers execution to the user application located at program memory address 0xC02. On the other hand, if the bootloader detects UART activity before it suspends itself, it programs program memories with the data it receives from the bootloader host application via UART interface.

The bootloader host application parses the HEX file containing the user application (generated by MPLAB® IDE) and sends this data to the bootloader target application via UART. The bootloader host application also supports additional features, such as read of program memories.

FIGURE 1: BOOTLOADER PROCESS



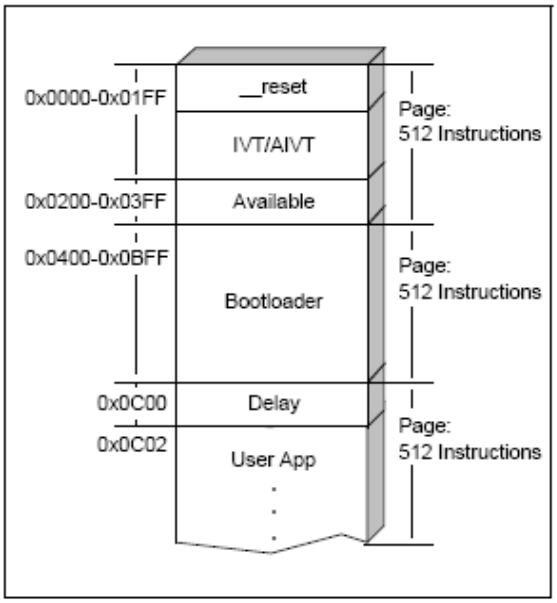
1. Bootloader host program reads user application.
2. Bootloader transfers user application to target device (via communication channel).
3. Bootloader target program loads user application into target device memory.

DEVICE MEMORY USAGE

Figure 2 illustrates memory organization for dsPIC33F targets. The interrupt tables (IVT/AIVT) use memory space up to address 0x1FE. The bootloader can not be placed immediately following this address because erasing the first Flash memory page also erases the bootloader. Therefore, the bootloader must start at address 0x400, which leaves a “hole” of unused memory from 0x200 through 0x3FE. However, this available memory can be used for your user application.

Also, because of this Flash memory page restriction, the user application can not be placed immediately after the bootloader. It must be pushed to the beginning of the next Flash memory page (address 0xC00). Starting at that address, the application specifies the bootloader delay value, followed by the actual application code at address 0xC02.

FIGURE 2: dsPIC33F AND PIC24F/24H PROGRAM MEMORY



DEVICE PERIPHERAL USAGE

The target side bootloader application uses program memory from address 0x400 to 0xC00 (inclusive) on dsPIC33F devices. It also uses Reset vectors from the Interrupt Vector Table (IVT). The target side bootloader application uses these peripherals:

- UART
- Timer

FILES

The bootloader application is organized into two subdirectories:

- Target Side: …\Bootloader\target
- Host Side: …\Bootloader\host

The target side bootloader application is developed with MPLAB IDE tools and consists of the following files:

- Project Files:
  16-bit Flash Programmer.mcp
  16-bit Flash Programmer.mcw
- Main Program (performs all main tasks, such as initialization and communication):
  main.c
- Support File (contains memory routines, such as erase and write):
  memory.s
- Test Application Files (located in the …\Bootloader\test directory)

The bootloader host application is developed with Python and consists of the following files:

bootloader.py.
bluetooth.py

BUILDING AND LOADING THE TARGET SIDE BOOTLOADER

To build and load the target side bootloader:

1. Open the project file: 16-bit Flash Programmer.mcp
2. Connect the target board to the host computer via MPLAB ICD 2.
3. From the **Programmer** menu, choose **Select Tool**, then click on **ICD2**.
4. From the **Programmer** menu, select **Program**.
5. Reset the target board.

At this point, the bootloader reads the delay value of 0xFF (since Flash was erased by the MPLAB IDE tools), and waits for UART activity.

TABLE 1: VALID DELAY VALUES

Delay Value (Seconds)	Results
0	Suspend bootloader and transfer execution to the user application.
1-254	Wait specified number of seconds for HEX file transfer. If no serial communication is detected before the delay time has expired, suspend bootloader and transfer execution to the user application.
255	Wait forever for HEX file transfer.

REQUIREMENTS FOR A USER APPLICATION

The following requirements apply to any application intended to be loaded by the bootloader:

- Application cannot place code into memory space reserved by the bootloader.
- Bootloader delay must be specified for subsequent bootloader executions.

To satisfy these requirements, the corresponding user application's linker script (.GLD file) must be modified to specify the application address and designate the bootloader delay period. For dsPIC33F devices, the .GLD file is modified to place the user's application at address 0xC02 and provide a time-out value for the bootloader. **It is strongly recommended to use the p33FJ128MC706.gld file located in the …\Bootloader\test directory**.

EXAMPLE

```
program (xr) : ORIGIN = 0xC00, LENGTH = 0x29E00
__CODE_BASE = 0xC00; /* Handles, User Code, Library Code */
/*
** User Code and Library Code
*/
  .text :
  {
       SHORT(0x05);    /* Bootloader timeout in sec */
       *(.init);
       *(.user_init);
       *(.handle);
       *(.libc) *(.libm) *(.libdsp);  /* keep together in this order */
       *(.lib*);
  } >program
```

Although configuration bits in user applications are ignored and not written to Flash memory, it is recommended not to add configuration bits in user applications. The following configuration bits are already included in the target device's bootloader.

```
_FOSCSEL(FNOSC_PRIPLL & IESO_ON);
_FOSC(POSCMD_EC & FCKSM_CSDCMD & OSCIOFNC_OFF);
_FWDT(FWDTEN_OFF); // Watchdog Timer Disabled
```

Configuration bits in user applications are redundant and they may cause serious problems.

Clock setup is not necessary since the following lines are already added in the target device.

```
_PLLDIV = 6;  // M = 8
_PLLPRE = 0;  // N1 = 2
_PLLPOST = 0; // N2 = 2
```

Clock switching and UART2 setup are already in the target device. Do not add them in user applications.

LOADING USER APPLICATION WITH THE BOOTLOADER

Use the following procedure to load the user application into the target device

1. Run bootloader.py. Figure 3 illustrates host PC bootloader.
2. Click on **Browse** and select the .hex file you want to download.
3. Power on the target hardware. Both red and green LEDs on the target hardware will on.
4. Click on **Connect** for the communication through Bluetooth between the PC host and the target hardware. If successful, the device name (dsPIC33FJ128MC706) will be displayed on the status bar as shown in Figure 3. The red LED on the target hardware will be off while the greed LED is still on.
5. Click on **Download** to start writing user application onto the target device.
6. If loading is successful, the standard output window will display results similar to Figure 4, and the user application will be running.

FIGURE 3: HOST PC BOOTLOADER

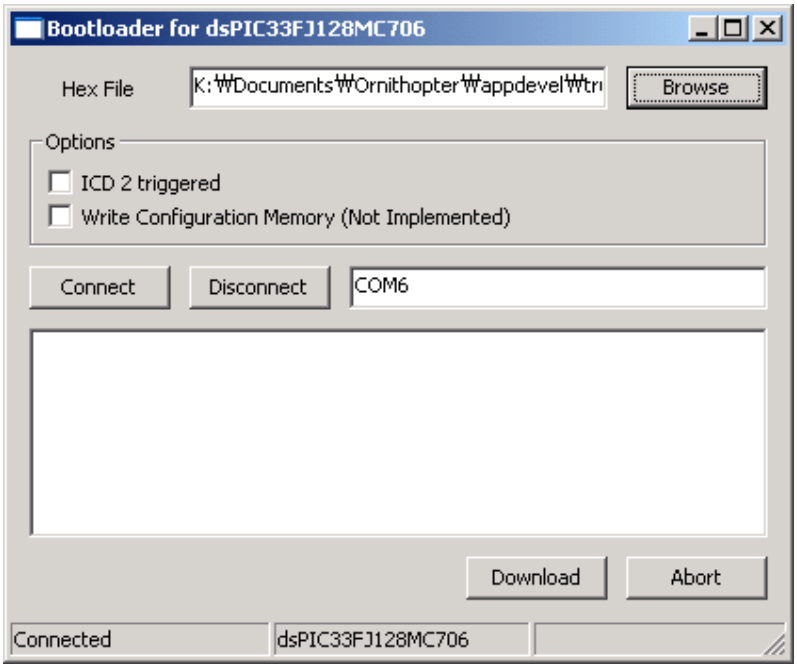
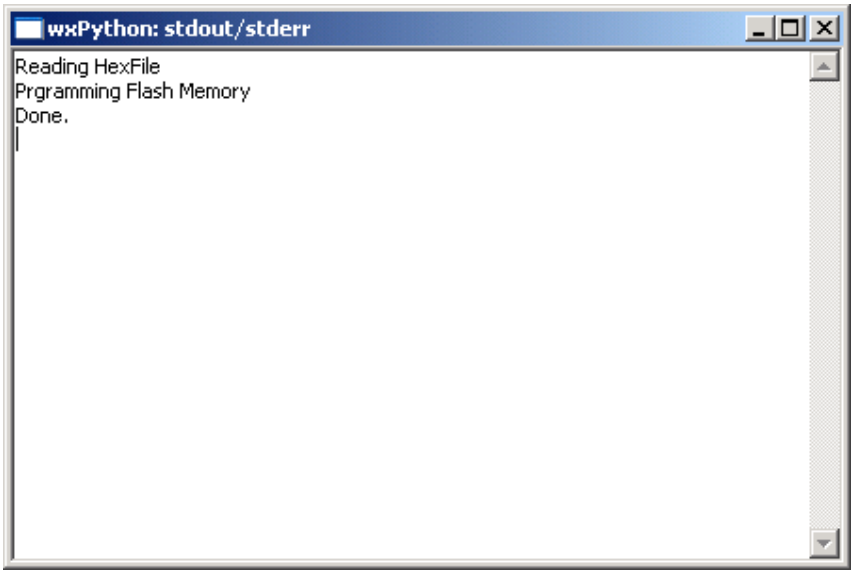


FIGURE 4:



RUNNING USER APPLICATION

User application can be started by one of the following three ways:

- Power on the target hardware and wait for the time specified at the address 0xC0.
- Power on the target hardware and send 0x00 to the target device via Bluetooth communication link.
- Power on the target hardware and run the handshake procedure (written in bluetooth.py module) followed by transmitting of 0x00.

Note: This document is based on AN1094 (DS01094A, Microchip), “Bootloader for dsPIC30F/33F and PIC24F/24H Devices.” I rephrased it as I needed.