ds30 Loader free edition
Main manual

DigSol Sweden AB

Table of contents

Introduction

ds30 Loader

ds30 Loader is a boot loader supporting PIC12, PIC16, PIC18, PIC24, dsPIC, and PIC32 families of MCUs from Microchip. It supports all devices in each family out of the box (those in production). The firmware is written in assembler. The PC clients run on Windows, Linux, and macOS.

Prerequisites and Requirements

Requirements for the different parts of ds30 Loader is found in their respective manual.

Trademarks

All rights to copyrights, registered trademarks, and trademarks reside with their respective owners.

Why use a boot loader

A boot loader usually allows software upgrade with cheap or generally available equipment such as an RS232 port, as opposed to specialized and expensive equipment such as a PIC programmer. Write time might also be lower with a boot loader.

Drawbacks of using a boot loader include added boot-up time and increased flash memory usage.

Usage

Operating system considerations

Windows

N/A

Linux

The user running ds30 Loader should be a member of the dialout group. To add a user in the dialout group run the following command as a super user:

```
useradd -G dialout username
```

macOS

The user running ds30 Loader should have sufficient rights to read from and write to the desired port.

Install the required tool suite

The required compiler/assembler tool suite needs to be installed before the boot loader firmware is opened in MPLAB X IDE. If the tool suite is not installed when the project is opened important settings may be lost. See the firmware manual for information about the required tool suite(s).

Modify firmware settings

Refer to the firmware manual for detailed information.

- Start Microchip MPLAB X IDE.
- Open the firmware by clicking Open project on the File menu, then browse to the ds30loader_xxx.X directory inside one of the firmware_xxx directories.
- Open settings.inc by double-clicking it in the project tree to the left. Review all settings and modify when needed.
- Open user_code.inc by double-clicking it in the project tree to the left. Review the available macros/functions and add init/exit code if needed.
- Use exactly the same configuration bits from the application in the boot loader unless you have a good reason not to and know exactly what you do.

Build the firmware

Build the firmware by clicking Build on the Production menu. Notice any warnings in the output window. Correct any errors that appear in the output window. Refer to the firmware manual for detailed information about ds30 Loader specific warnings and errors.

Write firmware

Write the firmware to your device using your favourite programmer. For detailed information, refer to the firmware manual.

Setup the host application

- Start ds30 Studio.
- Create a new project.
- Select the correct device on the project tab
- Select the correct boot loader on the project tab.
- Select the application hex file on the project tab.

- Configure the communication settings on the communication tab.
- Review settings in the boot loader tab.

See the ds30 Studio manual for more information.

Write the application

There are three ways to invoke the boot loader and perform the write.

Procedure 1

1. Press the write button in the GUI.
2. Reset the PIC/dsPIC.

Procedure 2

1. Reset the PIC/dsPIC.
2. Press the write button in the GUI before the boot loader times out.

This procedure may be more reliable than procedure 1.

Procedure 3

0. Implement a reset command in the application. See the reset part in the configuration section in this document for more information.
1. Configure the reset command under the communication tab in the GUI (advanced mode need to be enabled first).
2. Press the write button in ds30 Studio.

Wait for the write operation to complete

Firmware matrix

The following firmwares are available. For detailed information refer to the firmware manual.

	UART
PIC12	x
PIC16	x
PIC18	x
PIC24FK	x
PIC24FJ	x
PIC24H	x
dsPIC30F	x
dsPIC33F	x

Commercial version

The commercial version also support PIC24E, dsPIC33C, dsPIC33E, PIC32MM, PIC3MX. PIC32MZ, CAN, I^{2}C, SPI, SD Card and software UART.

Clients

ds30 Studio Console

A command line interface that runs on Windows. Refer to the console manual for more information. It may also run on Linux and macOS using the Mono framework.

ds30 Studio

It is a graphical user interface that runs on Windows. Refer to the GUI manual for more information.

Commercial version

For the commercial version .NET API, native API and native console are also available.

Configuration

This section is valid for ds30 Studio and ds30 Studio console.

Basic settings

Port name

The name of the port that will be used to communicate with the boot loader firmware.

Baud rate

The baud/bit rate that the port will be setup for prior to communicating with the boot loader firmware.

Write flash

When checked, the flash memory is written during a write operation. This is disabled if the chosen hex-file does not contain any program memory locations.

Write EEPROM

When checked, the EEPROM memory is written during a write operation. This is disabled if the chosen hex-file does not contain any EEPROM memory locations.

Read flash

When checked, the flash memory is read during a read operation.

Read EEPROM

When checked, the EEPROM memory is read during a read operation.

Serial port settings

Flow control

Sets the serial port flow control mode.

Parity

Sets the serial port parity mode.

Stop bits

One or two stop bits can be selected.

Advanced settings

Don't relocate application reset vector

When enabled, the first instructions in the hex file will not be moved to just before the boot loader. This can be useful when writing part of an application or only data to flash.

Don’t point reset vector to boot loader

Usually a goto to the boot loader is placed at 0x00. When this option is checked, the first few words at 0x00 will not be modified. If a hex file that contains data in the first page is written with this option enabled and the boot loader firmware does not feature goto protection, the boot loader will not work any more because it will not be called on start-up.

Don't write empty pages

When this is enabled pages that only contain 0xff will not be written. This may decrease write time and may also resolve the "The hex-file contains code that will overwrite the boot loader" message.

Allow overwrite of the boot loader

When a hex file is parsed there is control to make sure that the contents will not overwrite the boot loader, by enabling this setting that control is disabled. This does not bypass the firmwares own overwrite protection if any.

Custom boot loader

Allows proper operation with custom boot loader firmwares where size and placement has changed. Great care must be taken to set correct custom placement and size, they must match the BLPLP and BLSIZEP defines in the firmware. If a write is made with the wrong custom parameters, the boot loader may get broken if the firmware doesn't feature overwrite protection.

Write configuration bits

When checked, the configuration bits is written during a write operation. This is disabled if the chosen hex-file does not contain any configurations bits or if the hex-file contains more configuration bit locations than the selected device has. Writing configuration bits are only possible once per power on cycle.

One-way communications

The host application will insert a delay after sending a command instead of waiting for a response from the firmware.

Auto baud rate

When enabled, an auto baud rate synchronization character is sent prior to the write, read and check for boot loader operations. The firmware must be set-up for auto baud rate detection. Not all firmwares come with this feature.

Echo verification

If the device hardware or firmware is set-up to echo all received data this option must be enabled. When enabled, the ds30 Loader engine will compare all sent data with the echo and give an error if there is a mismatch.

Hello characters

Only for custom firmwares. Default hello character is 0xc1.

Timing

Hello timeout

Specifies the time to send the hello command before giving up when no response is received. If set to 0 there is not timeout.

Poll time

Specifies the interval at which the hello command is sent to the boot loader.

Timeout

Specifies the communication timeout.

Delay after port open

This delay is issued right after the port is opened. It can usually be kept at 0.

Reset

RTS

When enabled, the RTS pin is held high prior to communication with the firmware. The value of reset time specifies the time the RTS pin is held high.

DTR

When enabled, the DTR pin is held high prior to communication with the firmware. The value of reset time specifies the time the DTR pin is held high. Hardware support must be present.

Command

When enabled, a command is sent to the device to request reset or loading of the boot loader.

The reset time specifies the time to wait after the command is sent before trying to communicate with the boot loader.

The format of the data is hexval1;’string’;”string”….
Example hex only: 0;11;f;ab;3e
Example strings only: ‘Hello’;”Reset”
Example mixed: a5;’Reset’

Activation

RTS

When checked, the RTS pin is used to activate the device by holding it high during a write operation. Hardware support must be present.

DTR

When checked, the DTR pin is used to activate the device by holding it high during a write operation. Hardware support must be present.

Operation

Parsing of hex-file

The following operations take place during the parsing of a hex file:

Validation

The hex file is validated. Line checksums, file format and file completeness is checked. If any error is detected, the parsing is aborted.

Data collection

All data found in the hex-file that fits in the selected devices memory area are stored in RAM buffers.

Check of data that could overwrite the boot loader

If data found in the hex-file belong to the same memory space as the boot loader, a warning is displayed and the Write program check box gets disabled.

Check and move of GOTO

This step is not performed if the hex-file is encrypted. If no GOTO is found at address 0x00 the boot loader does not know how to load the user application and the Write program check box gets disabled. If a GOTO at 0x00 is found, it is moved to the two words just before the boot loader.

Counting of data

The data in the buffers are counted for presentation.

Set GOTO to boot loader

A GOTO to the boot loader is inserted in the first words beginning at 0x00.

Write

The following operations take place during a write operation:

Raise of process priority

If the poll time is set lower than 100ms, the GUI process priority is raised to above normal.

Device reset

If activated in the reset tab, the device is reset.

Auto baud rate

If auto baud rate is checked, the auto baud rate synchronization character (0x55) is sent using poll time until time out is reached or the boot loader responds.

Find boot loader

The hello command is sent using poll time until time out is reached or the boot loader responds. The boot loader responds with device id and firmware version. The received device id is checked against the selected device in the GUI.

Determine boot loader size

The boot loader size is determined based on the firmware version.

Parse hex-file

The hex-file is parsed if needed depending on if the determined boot loader size is different from the one used during last parse.

Write

Program, EEPROM and configuration bits are sent for write by the firmware. If checksum error is detected by the firmware, the GUI retries 3 times. If all 3 tries fail, the write operation is aborted.

Troubleshooting

Error and warning messages

Trouble	Possible cause	Solution
Verification failed	Broken chip	Replace the chip.
	Worn out flash	Replace the chip.
	Chip is used outside specifications	Read data sheet and make sure all specifications are met.
	Code protection is used	If code protection is used the device should be completely erased before the boot loader is written.
Uart rx buffer not empty as expected (is device sending data?)	This could happen if the device is not restarted properly and is sending data	Restart the device before pressing download.
Hardware detected a framing error	This might happen with serial ports when the baud rate error is just on the limit.	Try a different baud rate, higher or lower. Needs to be changed in both the boot loader firmware and the host application.
		Try another com-port.
The hex-file contains code that will overwrite the boot loader	This occurs when the loaded hex-file contains memory locations that could overwrite the boot loader. It can also occur if the wrong device is selected. Because the application does not know the actual boot loader size prior to communication with the boot loader it assumes a size. This value might be wrong; one can tell the application the correct value under the advanced tab.	Select correct device.
		Set custom boot loader properties under the advanced tab to the values from the firmware, usually called BLPLP and BLSIZEP in settings.inc.
		Free up program memory if needed.
		Reserve boot loader memory space in the application linker script.
Hex-file contains more config locations than the device has	If you use the export function from MPLAB this might happen.	Use the hex-file produced by the assembler or linker.
Last row containing configs was found in hex file, last page has been disabled. Consult manual for more information.	On PIC18FJ, PIC24FJ, and on some PIC24E/dsPIC33E devices the configuration words are stored as the last words in the ordinary flash address space. The configuration words are vital to the PIC operation. To avoid corrupting the configuration words or changing them which could lead to the boot loader not functioning properly the last page in flash will not get written.	To write the last page in flash, check the write configs checkbox under the advanced tab. It is unlikely that the last page will contain any code, therefore it is usually safe to not write the last page and this is also recommended.

There is no GOTO at location 0x00.	A custom boot loader is used where the application reset vector has been relocated.	Ignore the warning.

Appendix A – Links

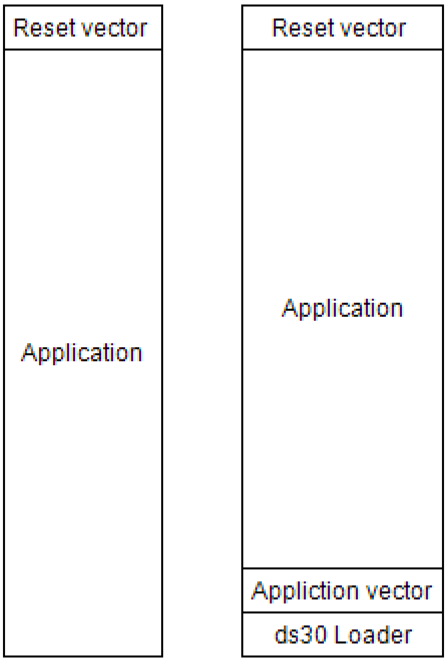
ds30 Loader website
https://ds30loader.com

ds30 Loader free edition website
https://picbootloader.com

Mono / MonoDevelop
http://www.mono-project.com

Appendix B – memory map

Memory maps without and with ds30 Loader.



Appendix C – Write time samples

These are sample times and should not be seen as a warranty, performance may be different in your application. Data is actual data that ends up in the flash memory.

UART	data [kB]	baud rate	time [s]	kB/s	Device
PIC12F	3,1	115 200	5,5	0,6	PIC12F1822
PIC16F	14,0	115 200	16,5	0,8	PIC16F1936
PIC18F	31,5	115 200	8,3	3,8	PIC18F2550
PIC18FJ	14,0	115 200	2,8	5,0	PIC18F24J11
PIC24F	15,5	115 200	2,3	6,7	PIC24F16KA102
PIC24FJ	125,0	115 200	15,5	8,1	PIC24FJ128GA010
PIC24H	126,5	115 200	15,8	8,0	PIC24HJ128GP504
dsPIC30F	47,3	115 200	7,7	6,1	dsPIC30F4011
dsPIC33F	254,5	115 200	31,9	8,0	dsPIC33FJ256GP710

Appendix D – Files

File	Description
bin\	**Contains binaries**
ds30Studio.exe	Graphical user interface for ds30 Loader
ds30StudioConsole.exe	Console application for ds30 Loader.
documentation\	**Contains all documentation for ds30 Loader**
firmware_xxx\	**Contains a MPLAB IDE project for one or more Microchip device families**
ds30Loader_xxx.X	MPLAB X IDE project directory
copying.txt	The license for ds30 Loader
ds30loader.mcp	MPLAB IDE project file
Ds30Loader.mcw	MPLAB IDE workspace file
src\devices*.inc	Contains device constants
src\settings.inc	Contains boot loader settings that should be checked/changed before download
uart.inc	
src\user_code.inc	User initialization and exit code
src\ds30loader.s/asm	Contains the boot loader code

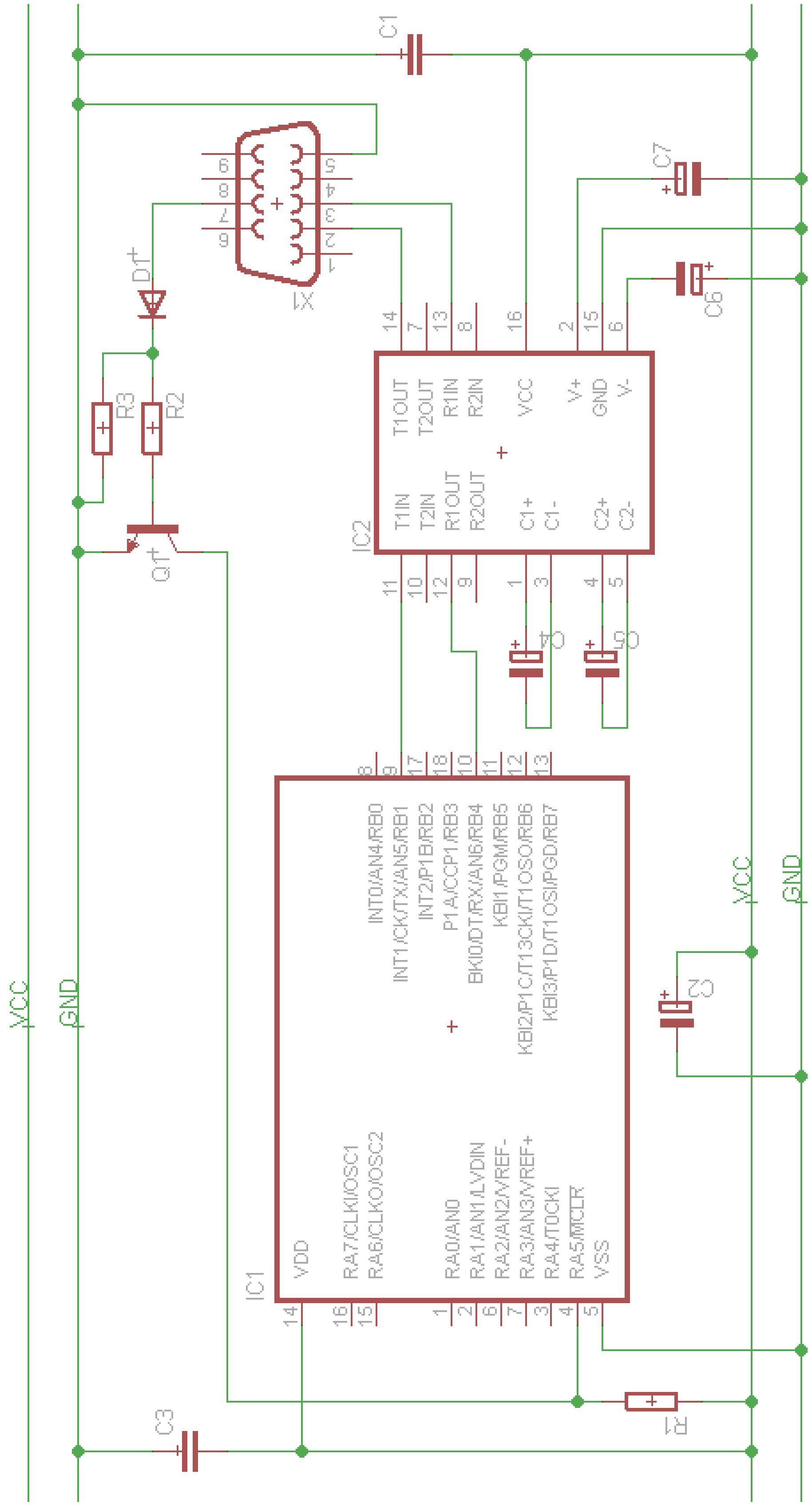
Appendix E – UART schematics

This is a sample implementation of UART communication. The schematic comes without warranty.

Components

Name	Value	Scope	Comment
IC1	PIC	Required	PIC microcontroller
IC2	MAX202	Required	RS232 level converter
X1	DB9	Optional	Connector
R1	10kΩ	Required	Master clear pull-up
C1	100n	Recommended	Decoupling capacitor
C2	10µF	Recommended	Filter capacitor
C3	100nF	Recommended	Decoupling capacitor
C4	100nF	Required	Charge pump capacitor
C5	100nF	Required	Charge pump capacitor
C6	100nF	Required	Charge pump capacitor
C7	100nF	Required	Charge pump capacitor
R2	10kΩ	Optional	Reset by RTS, base current limiter
R3	10kΩ	Optional	Reset by RTS, base pull-down
D1	1N4148	Optional	Reset by RTS, RTS signal rectifier
Q1	BC547	Optional	Reset by RTS, RTS signal inverter

Schematic


VCC
GND
C1
C7
C6
D1
R3
R2
Q1
X1
IC2
T1OUT
T2OUT
R1IN
R2IN
VCC
V+
GND
V-
T1IN
T2IN
R1OUT
R2OUT
C1+
C1-
C2+
C2-
C4
C5
IC1
INT0/AN4/RB0
INT1/CK/TX/AN5/RB1
INT2/P1B/RB2
P1A/CCP1/RB3
BKI0/DT/RX/AN6/RB4
KBI1/PGM/RB5
KBI2/P1C/T13CKI/T1OSO/RB6
KBI3/P1D/T1OSI/PGD/RB7
VDD
RA7/CLKI/OSC1
RA6/CLKO/OSC2
RA0/AN0
RA1/AN1/LVDIN
RA2/AN2/VREF-
RA3/AN3/VREF+
RA4/T0CKI
RA5/MCLR
VSS
C2
C3
R1