

**ds30 Loader**  
**Main manual**

## Table of contents

Table of contents.....	2
Document History .....	5
Introduction.....	6
ds30 Loader .....	6
Prerequisites and Requirements.....	6
Trademarks.....	6
Why use a boot loader .....	6
Usage .....	7
Operating system considerations.....	7
Windows.....	7
Linux .....	7
Mac OS X .....	7
Modify firmware settings.....	7
Write firmware.....	7
Prepare the application.....	7
Download the application to the device .....	8
Firmware matrix.....	9
Firmware matrix.....	9
Clients.....	10
Console.....	10
GUI.....	10
API.....	10
CAN.....	11
Peak.....	12
Lawicel.....	12
Configuration.....	13
Basic settings .....	13
Port name.....	13
Baud rate .....	13
Write flash .....	13
Write eeprom.....	13
Read flash .....	13
Read eeprom .....	13
CAN.....	13
Remote ID.....	13
Local ID .....	13
Extended frames .....	13
DLC.....	13
Advanced .....	14
Don't write goto to the user application.....	14
Allow overwrite of the boot loader.....	14
Write configuration bits .....	14
Custom boot loader.....	14

Auto baud rate .....	14
Echo verification .....	14
Put CRC before goto .....	14
Don't write empty pages .....	14
Don't set goto to bl at 0x00 .....	15
Timing .....	15
Hello timeout .....	15
Poll time .....	15
Timeout .....	15
Delay after port open .....	15
Reset .....	15
RTS .....	15
DTR .....	15
Command .....	15
ID .....	16
DLC .....	16
Baud rate .....	16
Activation .....	16
RTS .....	16
DTR .....	16
Security .....	16
Password .....	16
Operation .....	17
Parsing of hex-file .....	17
Validation .....	17
Data collection .....	17
Check of data that could overwrite the boot loader .....	17
Check and move of GOTO .....	17
Counting of data .....	17
Set GOTO to boot loader .....	17
Write .....	17
Raise of process priority .....	17
Device reset .....	17
Auto baud rate .....	17
Find boot loader .....	18
Determine boot loader size .....	18
Parse hex-file .....	18
Write .....	18
Boot loader upgrade procedure .....	19
Troubleshooting .....	20
Known issues .....	20
Error messages .....	20
HW schematics .....	22
UART / software UART .....	22
Components .....	22

Schematic .....	22
CAN .....	23
Components .....	23
Schematic .....	23
Appendix A – Links .....	24
Appendix B – memory map .....	25
Appendix C – Write time samples .....	26
Appendix D - Files .....	27
Package content .....	27
Files created by ds30 Loader .....	28

## Document History

### **February 2012**

The operations section was moved from the GUI manual

### **September 9th 2011**

New sections: firmware matrix and clients

Revised

### **Rev E**

Added appendix C write times

### **Rev D**

Memory map moved from firmware manual

Minor improvements

### **Rev C**

Added schematics

### **Rev B**

Added boot loader upgrade procedure

### **Rev A**

Initial release

## **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 Mac OS X.

### ***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
```

#### **Mac OS X**

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

### ***Modify firmware settings***

For detailed information, refer to the firmware manual.


- Start Microchip MPLAB IDE
- Open the firmware by clicking Open on the project menu, then browse to ds30loader.mcp which is located in the firmware directory.
- Open settings.inc by double-clicking it in the project tree to the left
- Modify or verify all values on lines marked with xxx
- Build the firmware by clicking Build all on the project menu
- Notice warnings on the output window
- Correct errors that appears in the output window

### ***Write firmware***

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

### ***Prepare the application***

You need to make sure there is a goto placed at location 0x00 in your application. In most cases your linker already does this for you. To check if it does you can pick Program Memory from the View-menu in MPLAB. It will look something like the screen-shot below. The GUI will let you know if it does not find a goto.

	Line	Address	Opcode		Disa
	1	0000	047F40	goto 0x007f40	
	2	0002	000000	nop	
	3	0004	007FEA	_DefaultInterrupt	
	4	0006	007FEA	_DefaultInterrupt	
	5	0008	007FEA	_DefaultInterrupt	
	6	000A	007FEA	_DefaultInterrupt	
	-	----	-----	- - - -	

### ***Download the application to the device***

- Start the GUI
- Select the desired communication settings and pick your hex-file. For details, refer to the GUI manual
- Press the download button
- Reset device manually if needed
- Wait for download to complete



## Firmware matrix

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

	UART	Sw UART*	CAN*	I <sup>2</sup> C*	SD card*
PIC12	X	X	n/a		
PIC16	X	X	n/a		
PIC18F	X	X	X		
PIC18FJ	X	X	n/a		
PIC24FK	X	X	n/a		
PIC24FJ	X	X	n/a		
PIC24H	X	X	X	see ds30 HEX Loader	see ds30 SD card loader
PIC24E	X	-	X		
dsPIC30F	X	X	X		
dsPIC33F	X	X	X		
dsPIC33E	X	-	X		
PIC32	X	-	X		

\* Only available in the commercial version

## **Clients**

### **Console**

It is a command line interface that runs on Windows, Linux, and MAC OS X. For more information refer to the console manual.

### **GUI**

It is a graphical user interface that runs on Windows, Linux, and MAC OS X. For more information refer to the GUI manual.

### **API**

The API can be use to integrate boot loader functionality and in a .NET application. It is ordered separately; contact MG Digital Solutions for more information.

## CAN

The ds30 Loader supports boot loading on the CAN bus. See table 1 for available firmwares and table 2 for supported CAN adapters. CAN operation is not available in the free edition of ds30 Loader.

Device Family	Firmware available	Supported DLC
PIC12	n/a	-
PIC16	n/a	-
PIC18	yes	1
PIC24F	n/a	-
PIC24FJ	n/a	-
PIC24H	yes	1-8*
PIC24E	yes	1-8
dsPIC30F	yes	1-8
dsPIC33FJ	yes	1-8*
dsPIC33E	yes	1-8
PIC32	yes	1-8

Table 1. Available CAN firmwares, \*depends on firmware type and version

Adapter	Supported bit rates	Windows		Linux		Mac OS X	
		x86	x64	x86	x64	x86	x64
IXXAT	10k, 20k, 50k, 100k, 125k, 250k, 500k, 800k, and 1M	X	X	-	-	-	-
Kvaser	50k, 62k, 83k, 100k, 125k, 250k, 500k, and 1M	X	X	-	-	-	-
Vector	100k, 125k, 250k, 500k, and 1M	X	X	-	-	-	-
Peak	5k, 10k, 20k, 50k, 100k, 125k, 250k, 500k, and 1M	X	X	-	-	-	-
Lawicel	10k, 20k, 50k, 100k, 125k, 250k, 500k, 800k, 1M	*	*	X	X	X	X
IS-CAN	10k, 20k, 50k, 100k, 125k, 250k, 500k, 800k, 1M	X	X	-	-	-	-
ESD	Contact						
IntrepidCS							

NI-CAN	
X = supported and recommended	* = supported - = not supported

Table 2. Supported CAN adaptors

### **Peak**

On x86 systems, PCANBasic\_x86.dll needs to be renamed to PCANBasic.dll

On x64 systems, PCANBasic\_x64.dll needs to be renamed to PCANBasic.dll

### **Lawicel**

There may be issues related to Lawicel CAN adapters, see the Trouble shooting-Known issues section in this document.

The Lawicel CAN232 adapter is shipped with the baud rate set to 57600. It can be increased for a performance improvement. It can be done in the configuration window available in ds30 Loader GUI. The serial interface baud rate can not be changed with the ds30 Loader console application.

If the ds30 Loader console application is used, the serial interface baud rate must be set manually in the Lawicel port configuration file to reflect the baud rate the Lawicel adapter is configured for. After a power cycle the CAN232 adapter defaults to 57600. If this is the desired baud rate, no action is necessary.

Adapters with older software does not support single filter mode which means in extended mode full filtering is not possible, only the 11 most significant bits are used to filter frames.

## **Configuration**

This section is valid for ds30 Loader GUI, ds30 Loader console and the ds30 Loader API.

### **Basic settings**

#### **Port name**

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

#### **Baud rate**

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

### **CAN**

#### **Remote ID**

This setting is only available for CAN ports, it specifies destination CAN ID.

#### **Local ID**

This setting is only available for CAN ports. It specifies local CAN ID. This can be entered as decimal (example 10) or hex (example 0xa).

#### **Extended frames**

When set extended frames are sent and only extended frames are received.

#### **DLC**

This settings determines how many bytes of data will be send in each CAN frame. Not that not all CAN firmwares support high DLC than 1.

## **Advanced**

This page contains advanced settings that are normally not used.

### **Don't write goto to the user application**

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.

### **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.

### **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.

### **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.

### **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.

### **Put CRC before goto**

Calculates a 16-bit (14-bit for PIC12 and PIC16) checksum value of the program memory and stores it just before the user application goto that in turn is placed just before the boot loader.

### **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.

### **Don't set goto to bl at 0x00**

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.

## ***Timing***

### **Hello timeout**

Specifies the time to send the hello command before giving up when no response is received.

### **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. This is only available for serial ports and hardware support must be present.

### **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. This is only available for serial ports.

### **Command**

When enabled, a command is sent to the device to request reset or loading of the boot loader. For syntax and examples of the reset command see Terminal Tx hex later in this document. When a CAN port is used, the macros \$PICID and \$GUIID can be used, they will be expanded to a two or four-byte big endian value depending on if extended is used or not.

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

## **ID**

The CAN id to be used when sending the reset command.

## **DLC**

The CAN DLC to use when sending the reset command.

## **Baud rate**

The baud rate to use when sending the reset command.

## ***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.

## ***Security***

### **Password**

The password will be sent before any boot loader operation is performed.



## 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.

## Boot loader upgrade procedure

This is not recommended and support is not guaranteed.

This guide assumes you have the ds30 Loader installed in the default location. The default location may differ between different firmware versions . Follow these steps carefully. The most critical step is 16: if it fails the boot loader is likely to stop working.

1. Make a copy of boot loader MPLAB project directory.
2. Open the MPLAB project
3. Open settings.inc
4. Disable goto protection if available
5. Enable boot loader protection if available
6. Change boot loader placement, BLPLP (this may be located in ds30loader.s/asm):

Device family	Boot loader placement of temporary boot loader, BLPLP/BLPLR
PIC16F	Upgrade is not supported
PIC18F	$BLPLP \times 2 + 1$
PIC18FJ	$BLPLP + 2$
PIC24F	$BLPLR \times 2 + 1$
PIC24FJ	$BLPLP + 2$
PIC24H	$BLPLP + 2$
dsPIC30F	$BLPLR \times 2 + 1$
dsPIC33FJ	$BLPLP + 2$

7. Build the project, Menu Project->Make or F10
8. Open ds30 Loader GUI and choose the hex file generated in the previous step
9. Download as you would with any hex file
10. If not already activated, activate advanced mod on the menu View->Advanced mode.
11. Check "Don't write goto at 0x00" under the advanced tab
12. Check "Custom boot loader" under the advanced tab
13. Put value of BLPLP/BLPLR from firmware in the placement textbox
14. Put value of BLSIZEP/BLSIZE from firmware in the size textbox
15. Choose the hex file of the new boot loader
16. Make sure the temporary boot loader is active, either using time or use a led to indicate that the temporary boot loader is active.
17. Download
18. Boot loader upgrade is finished

## Troubleshooting

### Known issues

- With the Lawicel CAN adapters read operation is not possible for most combinations of bit rate and serial interface baud rate. Typically any bit rate above 50k will not work for read flash operation. Write operation still works.

### Error messages

Trouble	Possible cause	Solution
"Verification failed"	Broken chip	Replace chip
	Worn out flash	Replace 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 when the baud rate error is just on the limit.	Try a different baud rate, higher or lower. Needs to be changed in both ends.
		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

“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 and PIC24FJ 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.	<p>To write the last page in flash, check the write configs checkbox under the advanced tab.</p> <p>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.</p>

## HW schematics

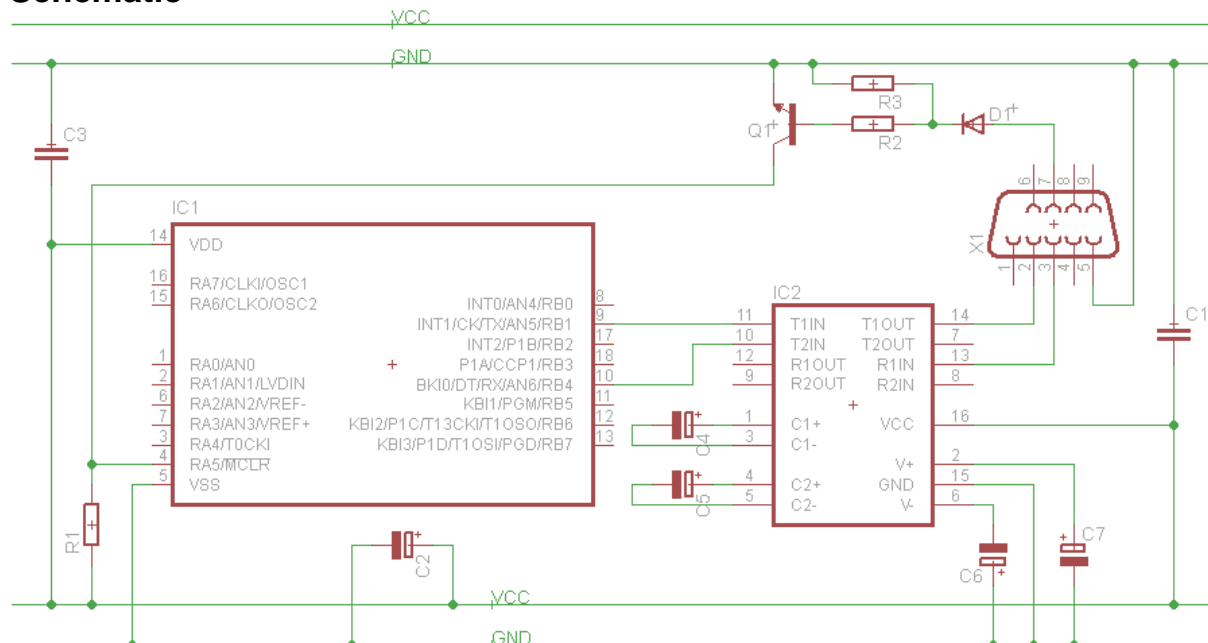
## UART / software UART

The schematic comes without warranty. Read datasheets and adapt the schematic to fit your specific application.

## Components

Name	Value		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



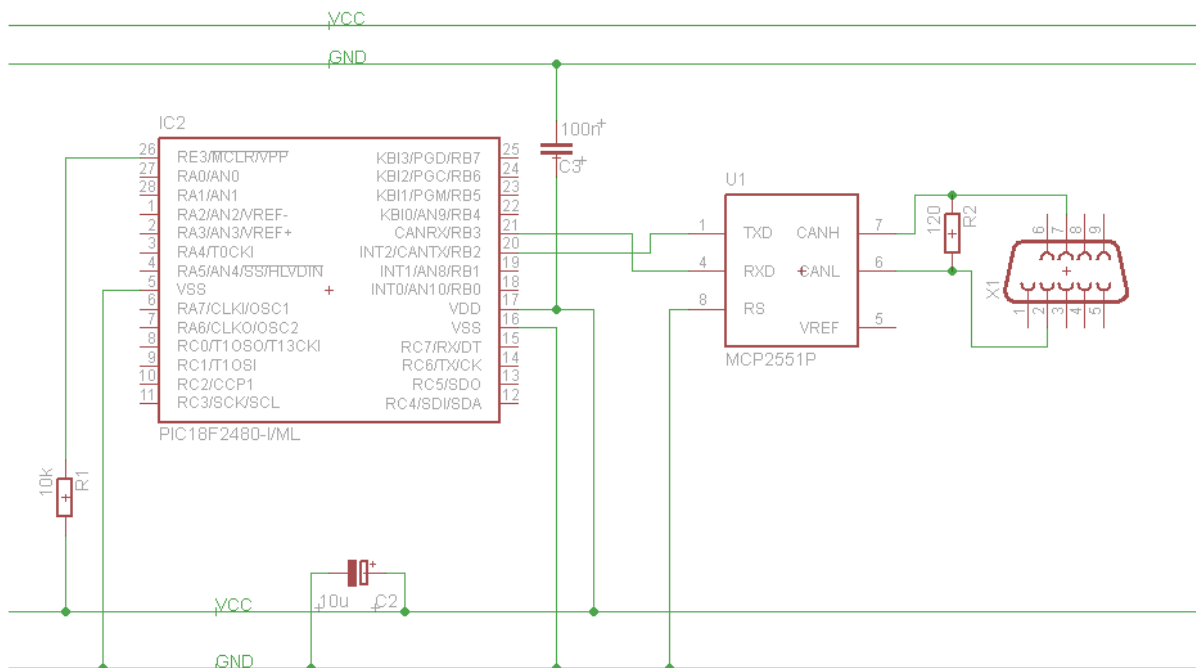
## CAN

The schematic comes without warranty. Read datasheets and adapt the schematic to fit your specific application.

## Components

Name	Value		Comment
IC1	PIC	Required	PIC microcontroller
U1	MCP2551	Required	CAN transceiver
X1	DB9	Optional	Connector
R1	10kΩ	Required	Master clear pull-up
R2	120Ω	Optional	Bus termination
C2	10μF	Recommended	Filter capacitor
C3	100nF	Recommended	Decoupling capacitor

## Schematic



## Appendix A – Links

ds30 Loader website

<http://www.ds30loader.com>

ds30 Loader free edition website

<http://mrmackey.no-ip.org/elektronik/ds30loader/>

Microsoft Visual C# express edition

<http://www.microsoft.com/express/vcsharp/>

Mono / MonoDevelop

<http://www.mono-project.com>

Eagle

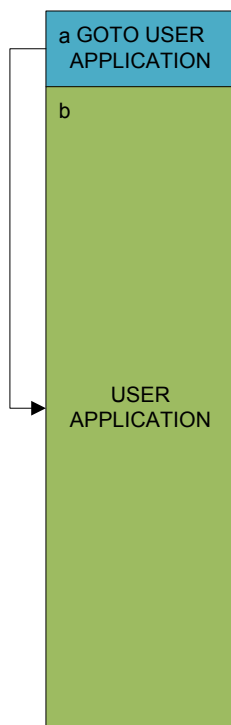
<http://www.cadsoft.de>



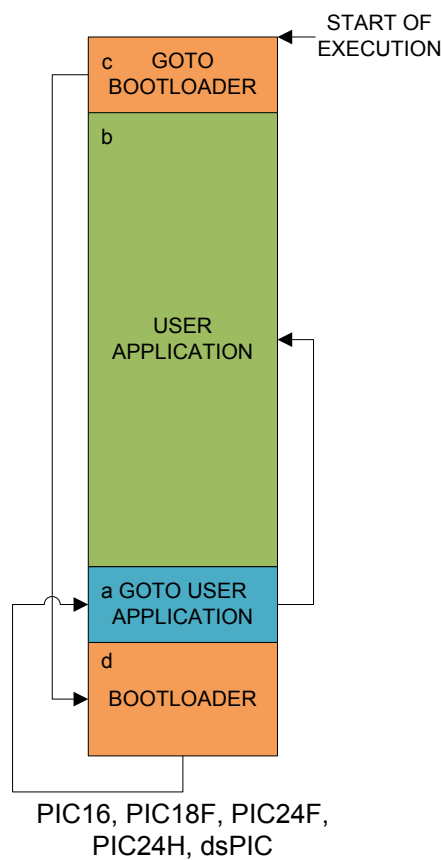
## Appendix B – memory map

- a. The goto application word at 0x00 is normally created automatically by the compiler. The GUI moves the goto to just before the boot loader code.
- b. The user application is not affected by the boot loader
- c. A new goto at 0x00 pointing to the boot loader code is created by the GUI.
- d. The boot loader code is normally placed at the end of the memory.

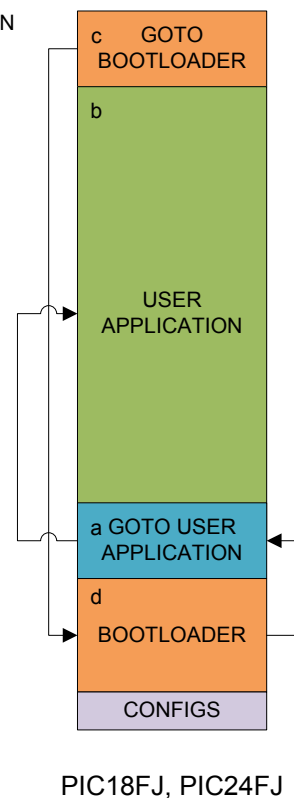
*Without bootloader*



*With bootloader*



*With bootloader*



## 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
PIC24HJ	126,5	115 200	15,8	8,0	PIC24HJ128GP504
PIC24E					
dsPIC30F	47,3	115 200	7,7	6,1	dsPIC30F4011
dsPIC33FJ	254,5	115 200	31,9	8,0	dsPIC33FJ256GP710
dsPIC33E	509,0	115 200	76,4	7,2	dsPIC33EP512MU810
PIC32MX	508,0	115200	69,4	7,3	PIC32MX795F512L

CAN	data [kB]	bit rate	time [s]	kB/s	Device
PIC18F					PIC18F66K80
PIC24HJ	126,5	250 000	40,0	3,0	PIC24HJ128GP504
PIC24E					
dsPIC30F	47,0	250 000	5,9	8,0	dsPIC30F4011
dsPIC33FJ	253,0	250 000	84,0	3,0	dsPIC33FJ256GP710
dsPIC33E					dsPIC33EP512MU810
PIC32MX	508,0	250 000	51,0	10,0	PIC32MX795F512L

## Appendix D - Files

### Package content

File	Description
\copying.txt	The license for ds30 Loader
<b>\bin</b>	<b>Contains binaries</b>
\bin\canlibCLSNET.dll	Use by the Kvaser CAN port plug-in
\bin\PCANBasic.dll	Used by the PCAN CAN port plug-in
\bin\PCANBasic_x64.dll	Used by the PCAN CAN port plug-in, this needs to be manually renamed to PCANBasic.dll on x64 systems.
\bin\PCANBasic_x86.dll	Used by the PCAN CAN port plug-in, this needs to be manually renamed to PCANBasic.dll on x86 systems.
\bin\vcinet2.dll	Used by the IXXAT CAN port plug-in
\bin\vxlapi.dll	Used by the Vector CAN port plug-in
\bin\vxlapi64.dll	Used by the Vector CAN port plug-in
\bin\vxlapi_NET20.dll	Used by the Vector CAN port plug-in Requires: vxlapi.dll
\bin\ds30LoaderGUI.exe	Graphical user interface for ds30 Loader
\bin\ds30LoaderConsole.exe	Console application for ds30 Loader.
\bin\ixxatBitRates.xml	Contains CAN timings used by IXXAT CAN port plug-in for different bit rates
\bin\vectorBitRates.xml	Contains CAN timings used by IXXAT CAN port plug-in for different baud rates
<b>\documentation</b>	<b>Contains all documentation for the ds30 Loader</b>
<b>\firmware xxx</b>	<b>Contains a MPLAB IDE project for a single Microchip device family</b>
\firmware xxx\ds30loader.mcp	MPLAB IDE project file
\firmware xxx\src\devices.inc	Contains device constants
\firmware xxx\src\settings.inc	Contains boot loader settings that should be checked/changed before download
\firmware xxx\src\user_init.inc	User initialization and exit code
\firmware xxx\src\ds30loader.s/asm	Contains the boot loader code
<b>\links</b>	<b>Contains links to web pages that are relevant to the ds30 Loader</b>
<b>\source</b>	<b>Contains a Visual Studio 2008 solution</b>
<b>\tools</b>	<b>Contains a few tools that may be useful when determining what settings one should use settings.inc</b>
\tools\dsPIC33 PLL settings.xls	An Excel sheet to help setting up the PLL on dsPIC33 devices. Provided by Leo Bodnar
\tools\UART error calculator	Helpful to calculate uart settings
\tools\CAN timing calculator	Helpful to calculate CAN settings

\\tools\\bldelay calculator.xls	Calculates timing constants
---------------------------------	-----------------------------

### ***Files created by ds30 Loader***

Created files are usually placed in user home directory\\ .ds30Loader

File	Description
settingsPortLawicel.xml	Settings saved by Lawicel CAN port plug-in
settingsPortSerial.xml	Settings saved by serial port plug-in
settingsPortVector.xml	Settings saved by Vector CAN port plug-in
settings.xml	GUI settings
recentfile*.xml	GUI settings for recent files