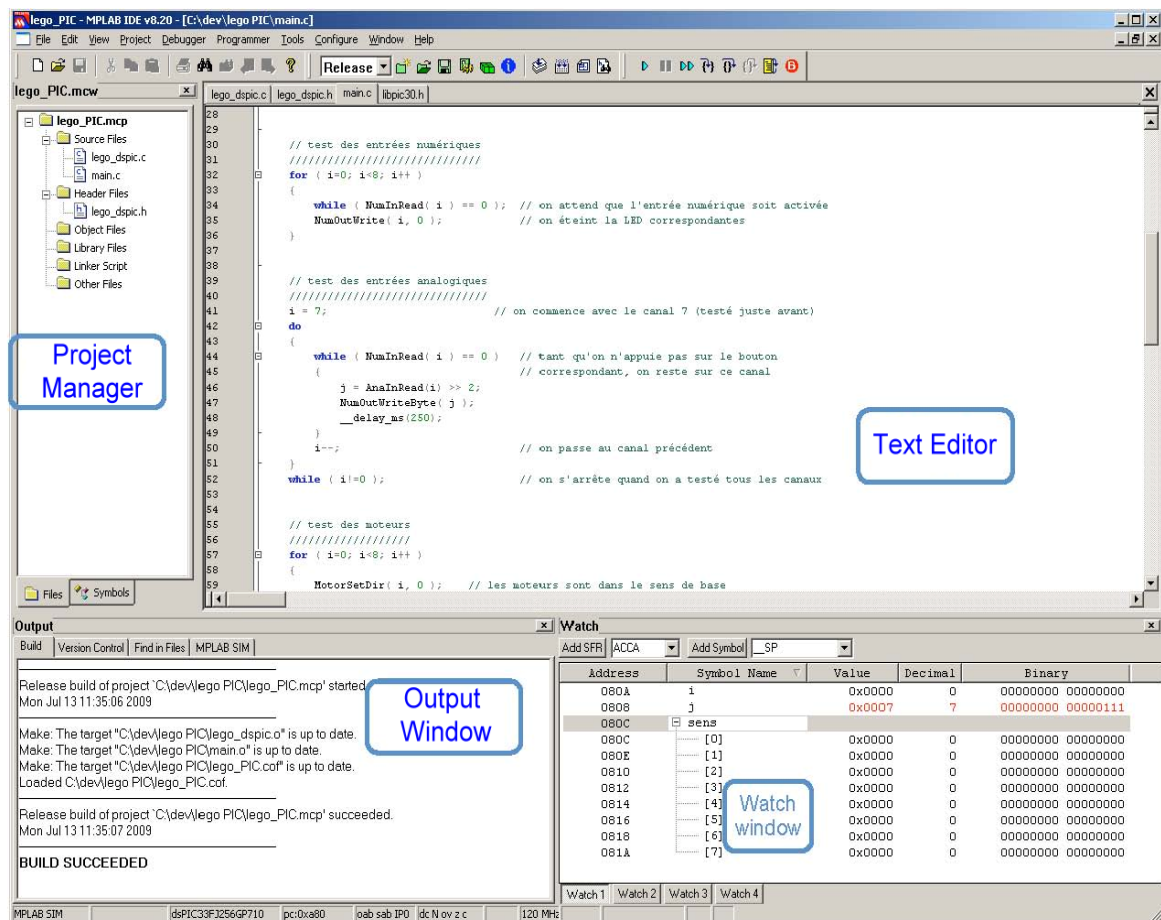


Introduction to MPLAB

Introduction

All the software tools required to the program of the PIC microcontroller family are gathered in an *IDE(Integrated Development Environment)* called *MPLAB*.

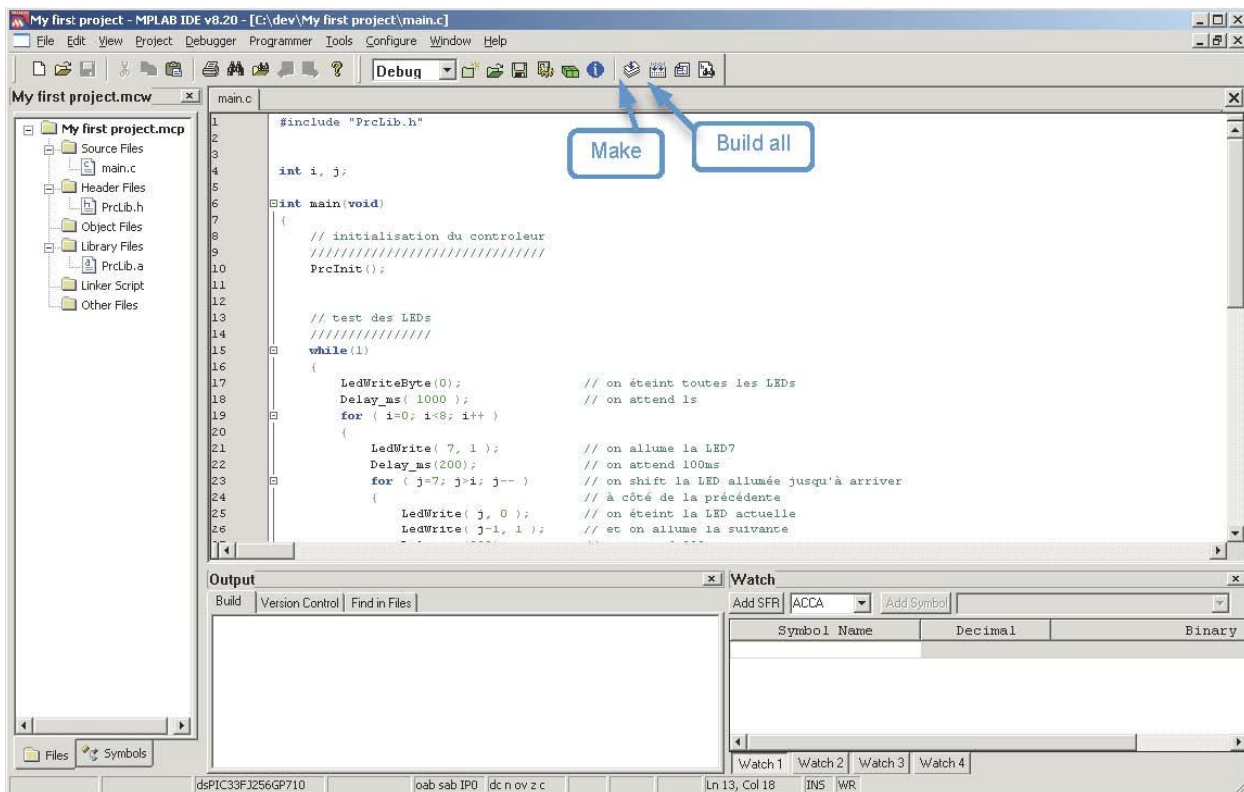
- a text editor, to write the source code; it is especially adapted to this task: the text is coloured to make it more readable, the indentation of the text (addition of tabulation to show the hierarchy of the loops) is automatic...
- a project, a manager which gathers all the files composing the project.
- an output window, in which the messages (information, warnings and errors) are displayed
- a "watch window", to display the state of the variables during the simulation or the debug of a program.
- toolbars and menus to launch the other tools: the compiler, the debugger, the simulator and the programmer.



Compile a project

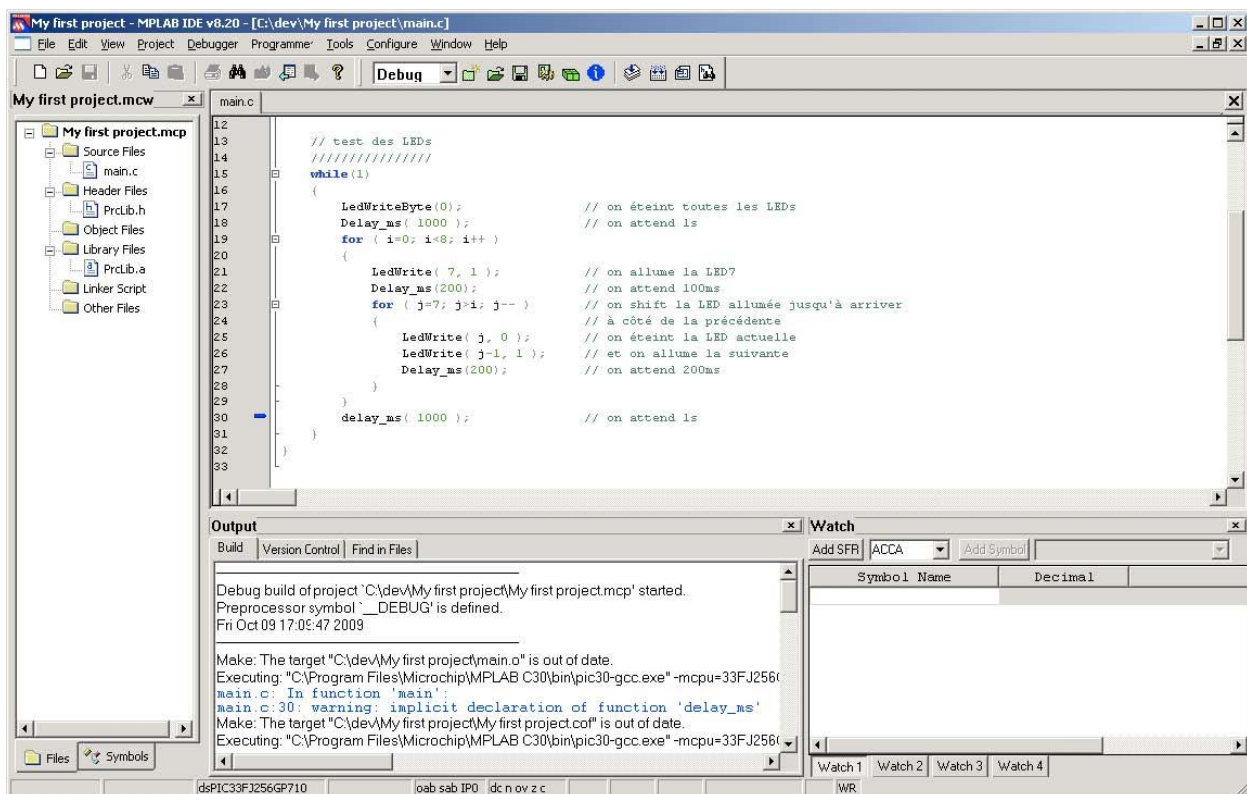
To compile your program, there are two commands:

- **Make** which compiles only the source files which were modified since last compilation. Therefore, the compiler compares the dates of the source files with that of the corresponding object file corresponding. **Make** saves time, chiefly for the projects comprising multiple files among which few are modified.
- **Build all**, which compiles all the source files of the project.



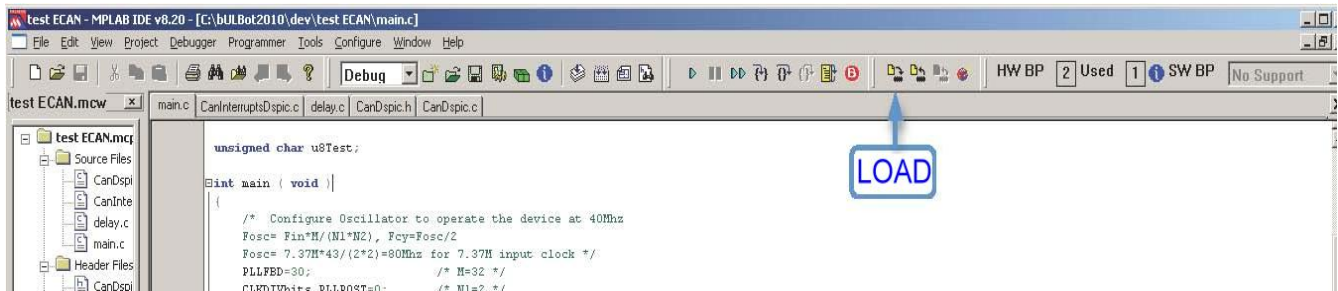
Correct the errors after the compilation

After compiling a project, we sometimes obtain errors or warnings. The compiler displays them in blue in the output window. Double-click on the error, the text editor jumps to the corresponding line in the source code, marked by a blue arrow in the margin. In the example below, *Delay_ms* begins with a lower case 'd', which provokes a warning during the compilation of the file *main.c*, and an error at the end of linking.



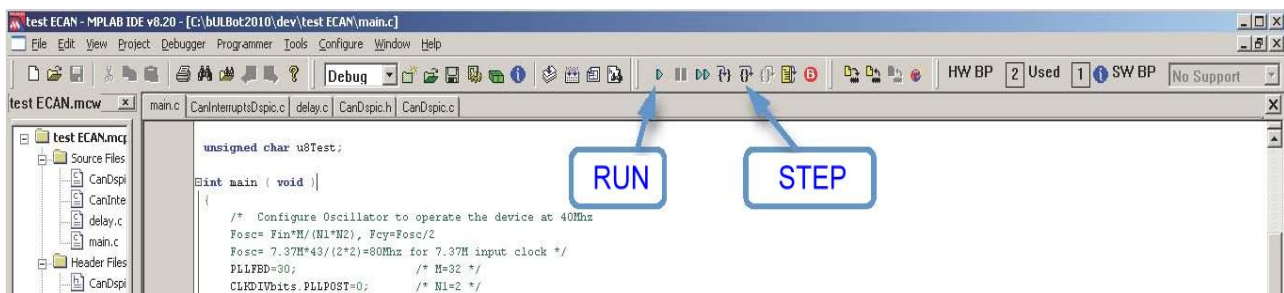
Load the program in memory

After the compilation, click on the "Load" button to transfer the machine code into the FLASH memory of the micro-controller



Run the Program

Click on the "Play" button to begin the execution and on "Pause" to stop at any time you can stop at any time. When in Pause you can display the state of the variables and of the registers in the Watch Window. You can also progress step-by-step to facilitate the debugging.



Use Breakpoints to debug the code

Repérer des erreurs de programmation n'est pas chose facile avec un microcontrôleur car les commandes du type *cout* et *cin* n'existent pas. C'est pourquoi l'on utilise très souvent les *breakpoints*. Le principe est le suivant : lorsque le programme exécute une ligne où se trouve un breakpoint, le processeur se met automatiquement en pause. Il est dès lors possible de visualiser l'état de toutes les variables et ainsi de vérifier le bon fonctionnement du programme. C'est également une manière simple de s'assurer que le programme exécute bien certaines lignes.

Track programming errors is much easier with breakpoints. The principle is simple: when the program executes a line where you have placed a breakpoint, the processor is paused automatically. Then you can visualize the state of all the variables and thus to check the correct operation of the program. It is also a simple way to make sure that the program actually executes a certain line

test ECAN - MPLAB IDE v0.20 - [C:\bULdot2010\dev\test ECAN\main.c]

File Edit View Project Debugger Programmer Tools Configure Window Help

Debug HW BP 2 Used 1 SW BP No Support

test ECAN.mcw

main.c CanInterruptDspic.c delay.c CanDspic.h CanDspic.c

test ECAN.mcp

- Source Files
 - CanDspic.c
 - CanInterruptDspic.c
 - delay.c
 - main.c
- Header Files
 - CanDspic.h
 - CanNetwork.h
 - common.h
 - delay.h
 - typedefs.h
- Object Files
- Library Files
- Linker Script
- Other Files

Files Symbols

```
// Map CAN pins to correct pins
RPIR26bits.C1RQR = 6; // Connect CANRX to RP6
RPIR3bits.RP7R = 16; // Connect CANTX to RP7

//*****
// Lock Registers
//*****
__builtin_write_OSCCONL(OSCCON | 1<<6);

AD1PCFGL = 15; // configure les pattes du CAN et des LEDs en pattes numériques

// Initialisations
CanInitialisation(4);

// declare quels objets on produit en CAN ainsi que leur ID
CanDeclarationConsumation(0x40, (void *)u8Test, sizeof(u8Test));
// Activation des interruptions
ACTIVATE_CAN_INTERRUPTS = 1; // Enables CAN interrupts

/* Infinite Loop */
while (1)
{
    CanEnvoiProduction(u8Test);
    Delay_Us(200); //On attend un peu pour éviter d'écraser le buffer de réception à la propulsion
}

return 0;
```

ACTIVE BREAKPT

Output

Build Version Control Find in Files MPLAB ICD 2

Auto-connect not enabled - Not connecting (Try enabling auto-connect on the ICD2 settings pages);
Connecting to MPLAB ICD 2
...Connected
Setting Vdd source to target
Target Device dsPIC33FJ128MC802 found, revision = Rev 0x3001
...Reading ICD Product ID

Watch

Acc SFR ACCA Add Symbol

Address	Symbol Name	Value	Decimal
040A	C1INTF	0x0000	0
0420	C1RXFUL1	0x0000	0