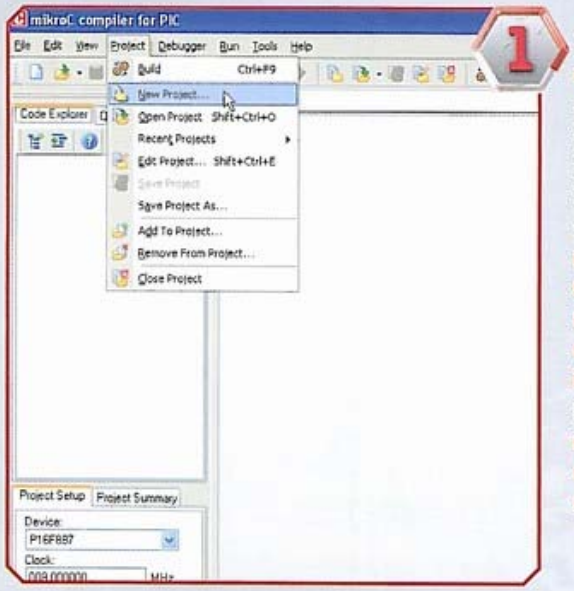
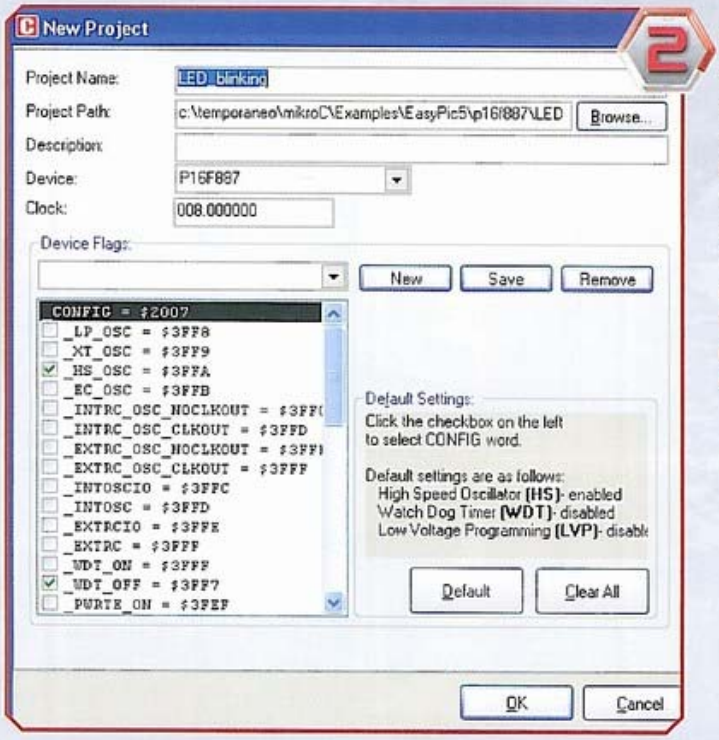
STEPbySTEP

CREIAMO IL NOSTRO PRIMO PROGETTO

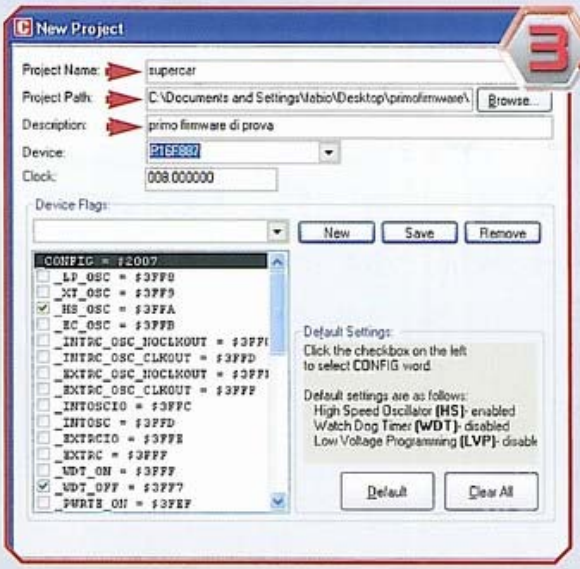
Nel Workshop precedente siamo giunti all'analisi del nostro primo firmware, che ci permetterà di implementare un semplice effetto luminoso. In questo fascicolo vedremo come compilarlo utilizzando mikroC.



Prima di ogni cosa avvia l'ambiente di sviluppo dal menu dei programmi di Windows. Una volta che il programma sarà pronto, dovrai creare un nuovo progetto. I progetti, all'interno degli IDE, rappresentano un modo per raggruppare e memorizzare sorgenti di codice, opzioni di configurazione e librerie di file, in modo da automatizzare i processi di compilazione. Per creare un nuovo progetto clicca sulla voce 'Project' del menu e successivamente sul comando 'New Project'.

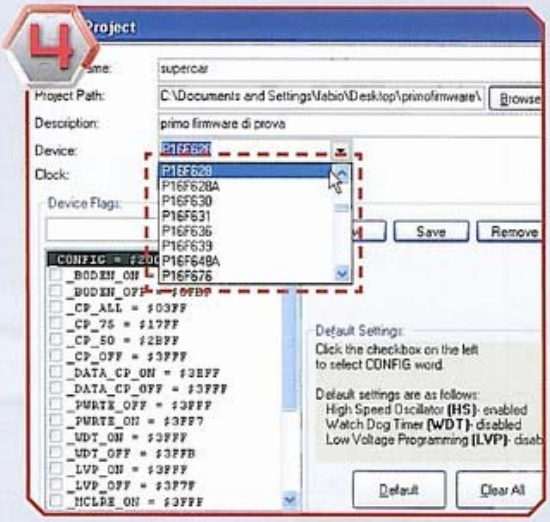


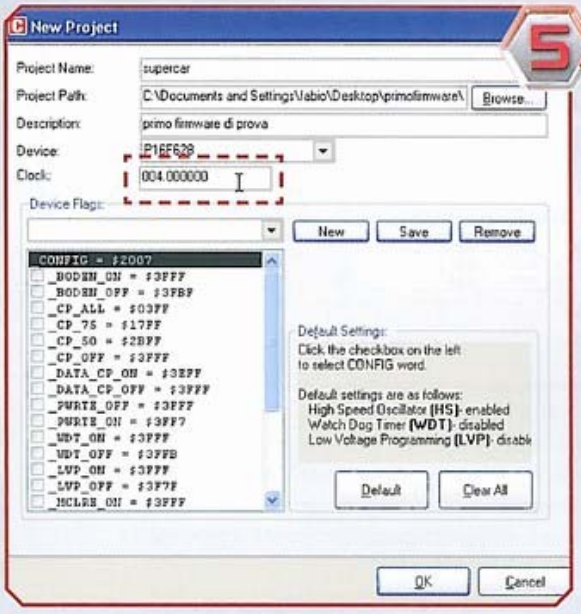
Dopo aver richiesto la creazione di un nuovo progetto sarà mostrata un'apposita finestra di configurazione, tramite la quale ti verrà data la possibilità di impostare l'ambiente di sviluppo secondo le caratteristiche del firmware che stai sviluppando.



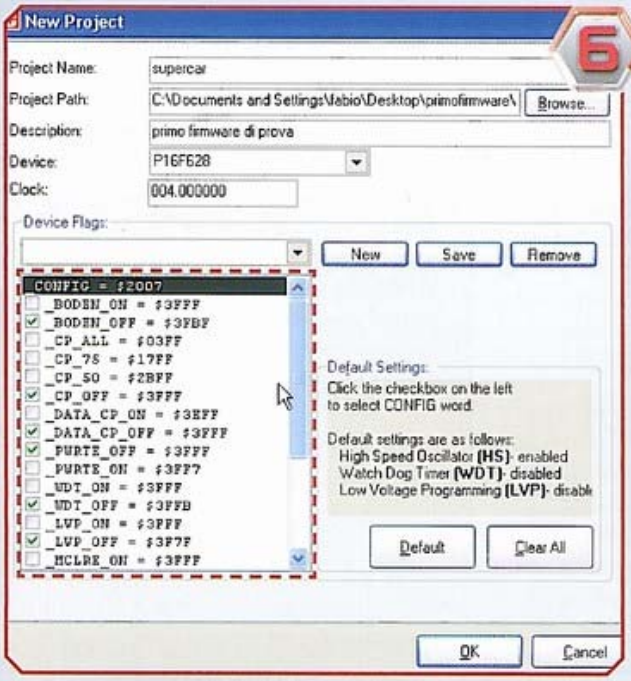
Inizia con il dare un nome al firmware e con l'indicare una cartella di salvataggio per il progetto. Per far ciò ti basterà inserire il nome scelto nel campo 'Project Name' e selezionare la cartella di salvataggio utilizzando il pulsante 'Browse...'. Se vorrai potrai inserire anche una breve descrizione del progetto.

Scegli dall'elenco 'Device' il PIC per cui vuoi sviluppare il firmware. Cerca 16F628 o 16F628A, in base al PIC che hai a disposizione (la versione A è un'evoluzione del 16F628 standard, con cui è comunque compatibile).





Una volta scelto il dispositivo indichiamo a mikroC che intendiamo utilizzare una frequenza di clock di 4 MHz agendo sul campo 'Clock'. Tieni presente che questo campo non ha nulla a che fare con la reale frequenza operativa del PIC, ma serve solo per parametrizzare correttamente alcune funzioni specifiche dell'ambiente di sviluppo, che richiedono informazioni sul clock. Nel nostro primo esperimento utilizzeremo il generatore di clock interno a 4 MHz.



6 **Arriviamo, infine, a configurare i 'fuse' del PIC, ossia i suoi parametri di funzionamento. Inizia impostando il primo insieme di parametri, come mostrato in figura. In questo modo il PIC sarà configurato in maniera seguente:**

_BODEN_OFF = con questa opzione disattivi il rilevamento del brown-out.

_CP_OFF = questa opzione ti permette di disattivare la protezione del firmware del PIC dalla lettura.

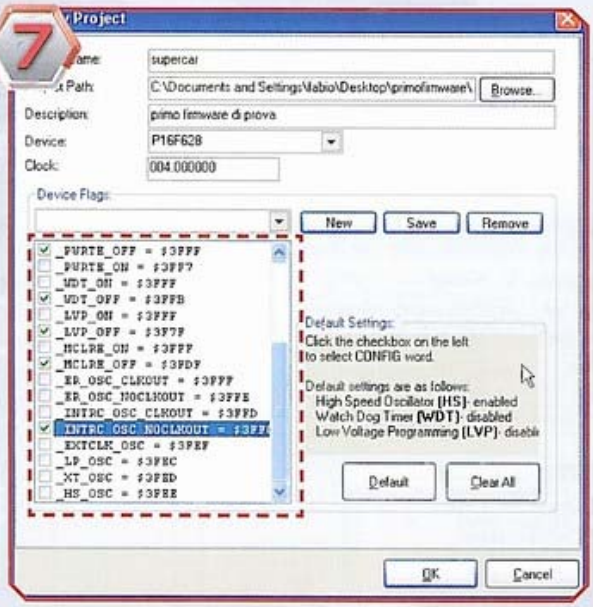
_DATA_CP_OFF = questa opzione ti permette di disattivare la protezione della EEPROM dati del PIC.

_PWRTE_OFF = disabilitiamo il timer di power-up.

_WDT_OFF = disabilitiamo il Watch-Dog Timer.

_LVP_OFF = disabilitiamo l'opzione di programmazione del PIC a basso voltaggio.

7 **Fai scorrere la lista dei fuse del PIC e completa la loro configurazione come mostrato in figura. Seleziona le ultime due opzioni di configurazione.**



_MCLRE_OFF = in questo modo disabiliti il funzionamento del PIC MCLRE (pin numero 4, MCLR/RA5), che consente di inserire un semplice circuito per il reset manuale del PIC. Selezionando questa opzione, il pin 4 sarà adibito a I/O digitale standard.

_INTRC_OSC_NOCLKOUT = questo fuse fa parte dei fuse di selezione del tipo di oscillatore. In particolare, ti permette di impostare il PIC in modo da operare utilizzando il suo generatore di clock interno (INTRC), senza inviare il clock generato in uscita sull'apposito pin del microcontrollore. Di seguito a questa opzione puoi vedere elencate le altre modalità di funzionamento che abbiamo incontrato nei fascicoli precedenti (EXTCLK, LP, XT, HS), che però richiedono componenti aggiuntivi.

Completata la configurazione clicca sul pulsante OK per creare il progetto.

supercar

```
1  void main()
2  {
3      int indice;
4
5      TRISB = 0b00000000;  //tutti i pin della porta B in output
6      PORTB = 0b00000001; //impostazione dello stato di uscita della porta B
7
8      while(1)
9      {
10       indice = 0;
11       //shift a sinistra
12       while(indice<3)
13       {
14         PORTB = PORTB<<1;
15         indice++;
16         Delay_ms(300);
17       }
18
19       indice = 0;
20       //shift a destra
21       while(indice<3)
22       {
23         PORTB = PORTB>>1;
24         indice++;
25         Delay_ms(300);
26       }
27     }
28 }
```

Il progetto è stato creato, non ti resta che copiare il sorgente mostrato nel fascicolo precedente all'interno dell'editor di codice di mikroC. Sulla sinistra puoi vedere sempre visualizzati i dati relativi alla frequenza di clock impostata e al microcontrollore di riferimento.

Compila il sorgente per verificarne il corretto funzionamento. Per far ciò clicca sulla voce di menu 'Project' e successivamente sul comando 'Build'. mikroC analizzerà e compilerà il sorgente. L'esito della procedura verrà visualizzato nell'area di output del compilatore (la tabella bianca nella parte bassa della finestra), all'interno della quale ti saranno fornite anche informazioni riguardanti la percentuale di memoria (memoria istruzioni

Messages | Find | QConvertor

Line/Column	Message No.	Message Text	Unit
0:0	100	Success (Release Build)	
0:0	101	Used ROM: 117 (5%)	Used RAM: 18 (8%)
0:0	102	Free ROM: 1930 (95%)	Free RAM: 206 (92%)

e RAM) 'consumata' dal firmware. Dopo aver compilato il sorgente, controlla la cartella di salvataggio del progetto. All'interno di essa troverai, oltre a tutti i file di progetto e i sorgenti C, anche un file con una particolare estensione '.hex'. Tale file è il firmware binario prodotto da mikroC. Tieni presente che, a differenza di quanto accadeva con Dev-C++, che produceva binari eseguibili a tutti gli effetti dal PC, questo file non è utilizzabile su un comune computer, a meno che non si usino appositi emulatori software del PIC specifico. Nel prossimo fascicolo vedremo la procedura che permette di caricare il firmware all'interno del PIC 16F628.