

Introduction to PROCESSOR EXPERT™

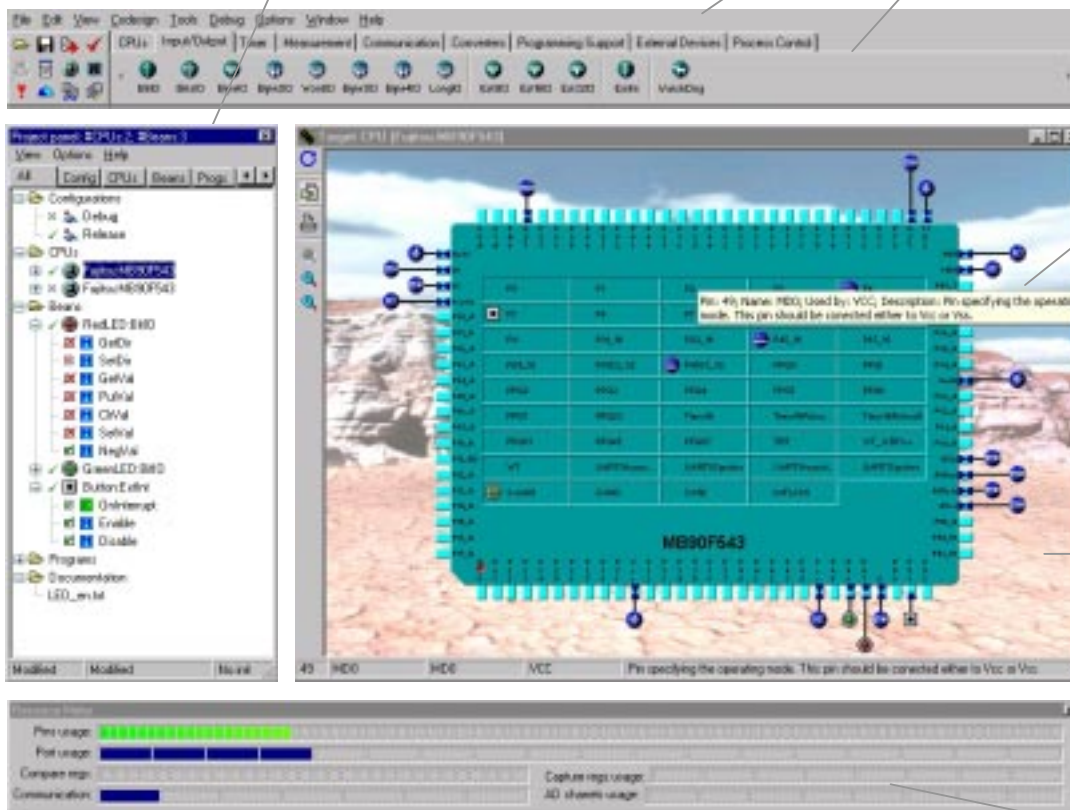
Rapid application design (RAD) tool for the codesign of microprocessors applications with Embedded Beans™

Project panel contains all the items related to the application, like CPU(s) Embedded Beans™, main program, etc. It allows quick access to Bean methods and events, code drag'n'drop, and localization into source files

Processor Expert™ main window consists of three parts

- main menu
- toolbar
- Embedded Beans™ palette, a palette that contains single Embedded Beans™

Embedded Beans™ encapsulate the functionality of embedded system basic elements like CPU core, CPU on-chip peripherals, standalone peripherals, virtual devices, and pure software algorithms, and change these facilities to properties, methods, and events (like objects in OOP).



Internal hardware knowledge expert system provides design time feedback for CPU resources.

This window displays the default CPU, its peripherals and pins, current data directions on the single pins and attached Embedded Beans™, pins reserved by user, busses etc.

Resource Meter shows actual status of used CPU resources on the given chip. Pins usage, port usage, compare registers, capture registers usage, serial usage, AD channels usage

Thanks to Processor Expert™, you no longer need to ...

- ...lose time in your data books, Processor Expert knows them all, and he is always there when you need him.
- ...be afraid of using an unfamiliar microprocessor. Processor Expert will make you an instant expert.
- ...be worried about changes in hardware/software implementation. Processor Expert handle them in no time.
- ...be stressed by time, safety and performance issues of coding. Processor Expert composes for you instantaneously optimized code.

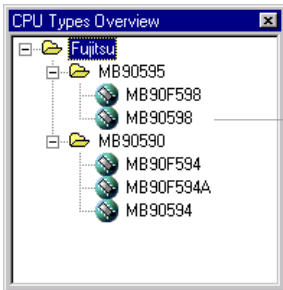
And you may now ...

- ...explore every alternative microprocessor and hardware/software architecture
- ...make full use of the features and performance of your microprocessors
- ...be the first on the market with the best product !

Processor Expert™ ... because tomorrow's products can't be designed with yesterday's tools

CPU Parameters Overview, #CPUs: 2 of 5															
CPU type	producer	family	clock	dual clock	temperature	#pins	#I/O ports	#timers	#A/D	#serial	watchdog	RAM	ROM	EPROM	FLA
MB90F594	Fujitsu	MB90590	4 to 16 MHz	no	-40 to 85°C	100	14	23/60/6	8	3/4	yes	6 KB	0 B	0 B	256
MB90F594A	Fujitsu	MB90590	4 to 16 MHz	no	-40 to 85°C	100	14	23/60/6	8	3/4	yes	6 KB	0 B	0 B	256

CPU Parameters Overview shows the list of CPUs and derivatives. Next to the CPU names you can also see a summary of their basic resources.

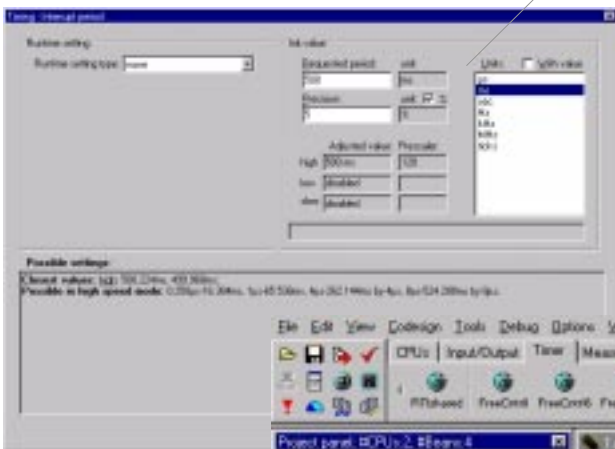


CPU Types Overview is a tree structure that gives an overview of the CPU producers, CPU families, and CPU variants. By selecting the CPU variant, you may view the chip in the Target CPU window.

General **Bean Inspector** unifies the access to properties, methods, events, comments, and other Embedded Bean features. You can see that CPUs are also **Embedded Beans**™



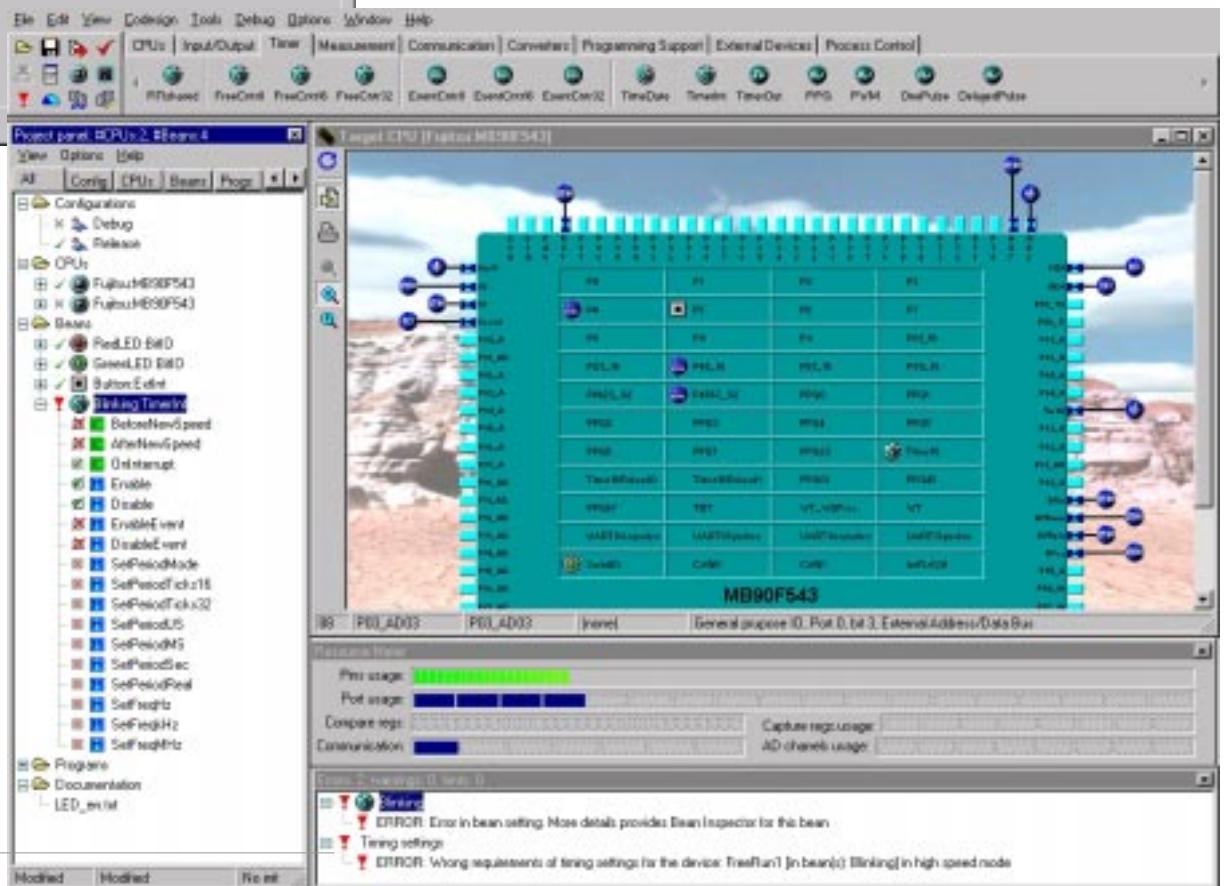
Even without knowledge of the current chip, you can make settings properly, using dialog provided by the Expert knowledge system. This is the first time that a development tool verifies your chip settings at design time, adding more safety at application runtime and less effort for debugging.



Frequently used beans can be customized with your settings and saved as **templates** that you may later add to a toolbar

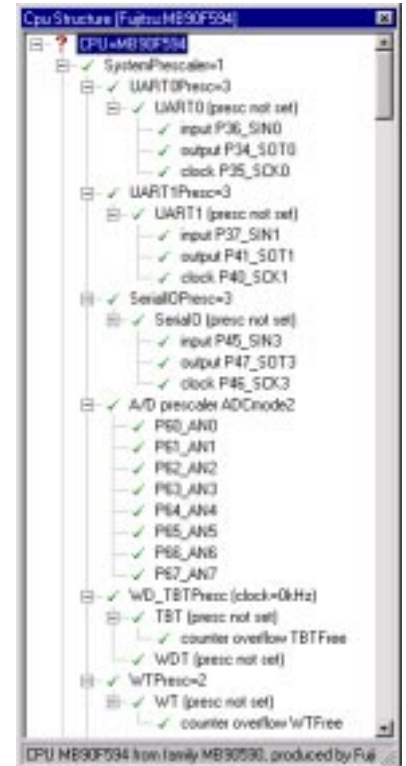


This window displays **errors, warnings, and hints** which are found during project design. Some errors are found right after inconsistent or incorrect data has been entered, others during the generation of a project. The single messages mention the bean where the error was found. If an error concerns two beans (where conflict results, for example, from using the same on-chip peripheral), the error will be attributed to both beans.

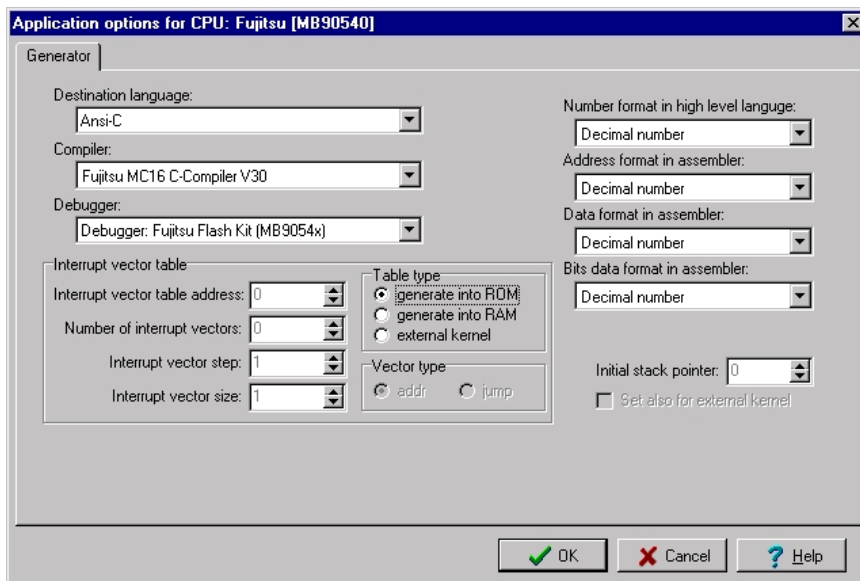




The CPU structure is presented schematically, in order to be easily and quickly visualized. The CPU device hierarchy appears as a tree structure, consisting of several main branches: **SystemClockPrescaler, I/O Ports, External Interrupts, DMA channels**



Extensive online HELP for each Embedded Bean™ is provided with HTML-based documents



Application Options are options for the Processor Expert™ code generator. They are local, i.e. they are valid only for the current application (an application is the subset of the project that concerns a given target processor). This approach offers advantages for developing the same project for many target CPUs. In the application option windows, we can choose the destination language and compiler from a list, and set up all other code generation options for the current application.



Process Control beans are Embedded Beans™ that implement a wide range of function modules for industrial applications. Most of them are designed for closed loop controlling, but many beans for standard control applications are also included. Like all other beans, they have properties, methods and events.

Processor Expert™ works as a virtual developer ...

```

File Viewer: codeassigned code Fujitsu.C
-----
Method      : Fujitsu_SetHighSpeed [EFP0000]
Description :
    Set high speed mode
Parameters  :
    None
Returns    :
    -----
    -Error code
-----
*/

```

Processor Expert™ works as a virtual developer and generates all the application code (application skeleton) automatically.

Here is the code for the CPU bean. The functionality of the CPU like any other elements of the system is expressed in terms of properties, methods, and events. This example shows the methods implementing the initializing functions.

```

Assigned class Fujitsu_SetHighSpeed()
-----
if (CpuMode != HIGH_SPEED) {
    CSCR_MCS = 1;
    __asm("LCSOP");
    CSCR_CSD = 1;
    CSCR_CST = 0;
    __asm("LCSOP");
    CSCR_MCS = 0;
    do {} while(CSCR_MCS);
    CpuMode = HIGH_SPEED;
}
return EFP_OK;
-----
*/

```

```

File Viewer: codeassigned code LEDa.C
-----
Description :
    This method returns the specified bit of the input value.
    a) direction = Input : reads the input value from pins
    and returns the specified bit
    b) direction = Output : returns the specified bit
    of the last written value
Parameters  :
    BitMask - Mask of the bit to read (0 to 7)
Returns    :
    -----
    - Value of the specified bit (FALSE or TRUE)
    FALSE = "0" or "Low", TRUE = "1" or "High"
-----
*/
byte LEDa_GetBit(byte BitMask)
{
    byte Mask=LEDa_GetMask(BitMask);
    byte Val;
    byte Result=0;
    if (Mask) {
        Val = PDM;
        Val=Mask;
        Result=(Val==Mask);
    }
    return(Result);
}
-----
*/

```

Here is the code generated for the ByteIO bean (note the frequency and quality of code comments). The example shows the code generated for the GetBit method of the ByteIO bean.

```

File Editor: Events.C
-----
Events LCDprj
**
** Event      : TmrOnInterrupt (module Events)
**
** From bean  : Tmr [TimerInt]
**
** Description :
**
** This event is called periodically with specified
** period/frequency (only when the bean is enabled - "Enak"
** and the events are enabled - "EnableEvent").
**
** Parameters :
**
** None
**
** Returns    :
**
** Nothing
**
** -----
**/
void TmrOnInterrupt(void)
{
    if (ShownPic1) {
        LCD_Put_image( 0, 0, Pic2_GetImage() );
        ShownPic1=0;
    } else {
        LCD_Put_image( 0, 0, Pic1_GetImage() );
        ShownPic1=1;
    }
}
}
-----
*/

```

In conclusion, the on-event code and main code to drive the system are added.

```

File Editor: LCDprj.C
-----
Events LCDprj
**
** UNIS s.r.o.
** Jurovskaya 22
** 624 00 Brno
** Czech Republic
**
** http   : www.processorexpert.com
** email  : expert@pauk.cz
**
** =====
**
** NOBLE LCDprj */
**
** Including used modules for compiling procedure*/
#include "C:\PE\FULL_F-1\Projects\LCD\OSES\Fujitsu.h"
#include "F:\PE_FUJ\Projects\LCD\OSES\Cpu.h"
#include "C:\PE\FULL_F-1\Projects\LCD\OSES\Events.h"
#include "C:\PE\FULL_F-1\Projects\LCD\OSES\LCD.h"
#include "C:\PE\FULL_F-1\Projects\LCD\OSES\Tmr.h"
#include "C:\PE\FULL_F-1\Projects\LCD\OSES\Fan1.h"
#include "C:\PE\FULL_F-1\Projects\LCD\OSES\Fan2.h"
**
void main(void)
{
    doc(1);
}
/* END LCDprj.C */
**
** =====

```

How to build support for embedded systems?

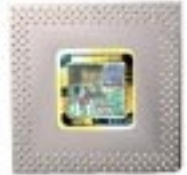
Graphical Display
LCD



EPROM



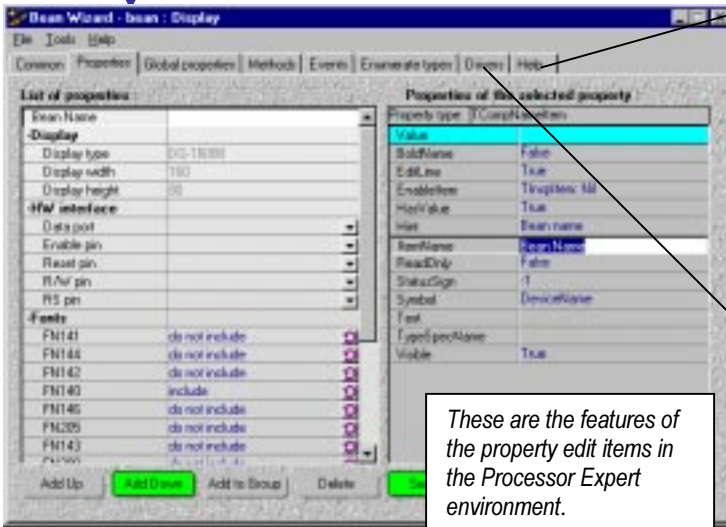
Intelligent Chip



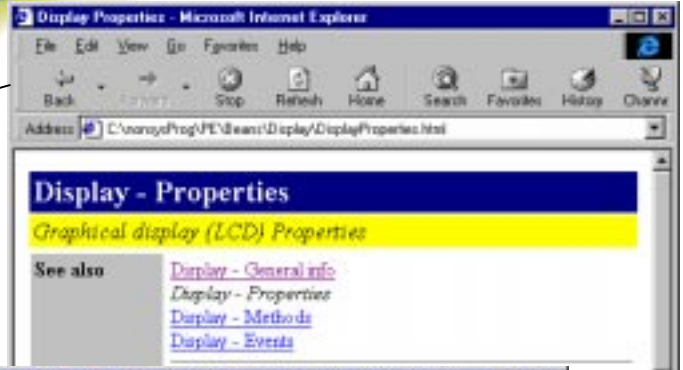
Software



In order to provide support to the user of this hardware component, a new Embedded Bean™ needs to be created. Bean Wizard helps the generation of new beans and facilitates the reusability of existing Beans (inheritance).



These are the features of the property edit items in the Processor Expert environment.



Once the new bean is generated, it can be added to the Bean Palette of Processor Expert. The Bean Palette Editor allows you to fully customize the Bean Palette.

