

PROFINET for embedded Systems:

- 32 Bit MCU
  - 96kByte RAM
  - on-board MAC
  - optional: external MAC
  - 1 Interrupt
  - 1 Timer
  - No Operating System (OS) needed, but supported on request
- Nothing more needed.



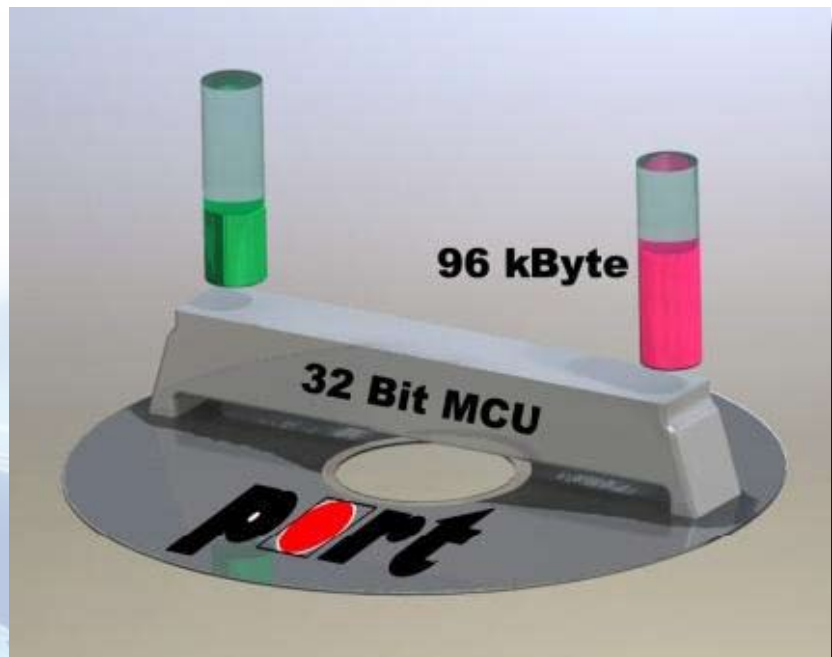
## port PROFINET Protocol Library

Port's PROFINET offers PROFINET 2.2 compatible communication not only for Embedded Systems. Supported are Conformance Class A (CC-A) and the Realtime Class 1 (RT-1).

A small driver application manages the adaption of the hardware and (if needed) to the Operating System. This approach reduces the "porting" to a new hardware platform or Operating System to a minimum.

The PROFINET stack operates in Embedded Systems mostly without any OS, however it can be connected to almost any (embedded) OS. Using an embedded 32Bit CPU, like e.g. STM32F207, cycle times of 1ms can be reached.

The use of the PROFINET stack is not limited to Embedded Systems, also powerful CPUs and the corresponding OS versions are supported. Here the PROFINET Stack is used as a regular user-application. The by the OS provided TCP/IP Stack can be used.



### Manufacturer

STMicroelectronics  
Texas Instruments  
Texas Instruments  
Texas Instruments  
Microchip  
Renesas  
Renesas  
Xilinx  
Xilinx  
Linux  
Windows

### Type

STM32F207, STM32F407  
LM3S9B92  
TIVA LM4xx  
Sitara AM335x SoC  
PIC32MX795  
RX63N  
R-IN32M3  
Zynq SoC  
Xilinx Spartan 6 on MicroBlaze  
Linux  
Windows 7 Prof. (Desktop-Line)

*Other platforms or*

*/and operating systems Possible, on request*



**Technical Details:**

The PROFINET Protocol Stack offers services according to the IEC-Standards IEC 61158 and IEC 61784. It enables the user for quick and reliable device development.

The hardware access is managed by a driver application via a defined interface. Drivers for various CPUs and Ethernet MACs with or without OS support are available. The Ethernet drivers are optimized for best performance.

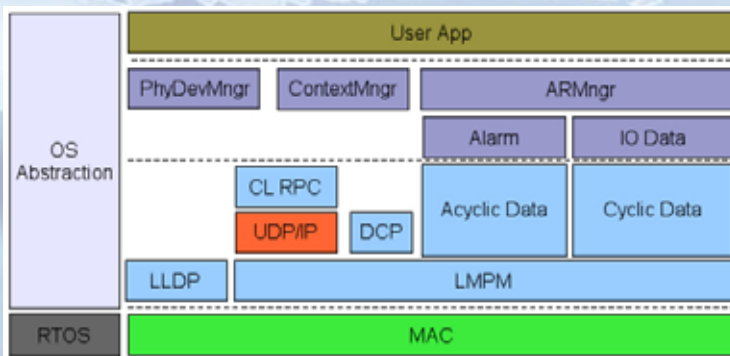
The PROFINET Protocol stack is written in ANSI-C and can be processed with any ANSI-C compliant compiler applications.

The user's application interacts with the protocol stack via function calls and call-back functions.

Devices, communicating based on port's PROFINET protocol stack can be certified by the PNO. While the stack provides all measures for certification, the user's application still must use the stack in the proper way.

As a part of the stack the  $\mu$ P TCP/IP protocol stack (licensed under a BSD like license) is provided. The TCP/IP Stack's RAM consumption is already included in the 96kByte total of RAM consumption. The  $\mu$ P stack can be swapped out for another stack according to the user's needs. The effort varies.

The structure of the stack is shown in the following schematic drawing:



**Table Showing:**

Feature	Support
Specification	2.2
Conformance Class	A
Realtime Class	1
PROFINET IO Device	Yes
PROFINET IO Controller	No
IRT Support	No
State Machine	Yes
Object Dictionary	Yes
Record Data	Yes
IO Data	Yes
Diagnose	Yes
Alarm	Yes
Isochronous Mode	No
Physical Device Manager	Yes
number of Application Relations (AR)	2 (1 Controller, 1 Supervisor)
number APIs	1 (API 0 manufacturer specific area)

**License model:**

Product license or Project license, No royalties.

**Delivery:**

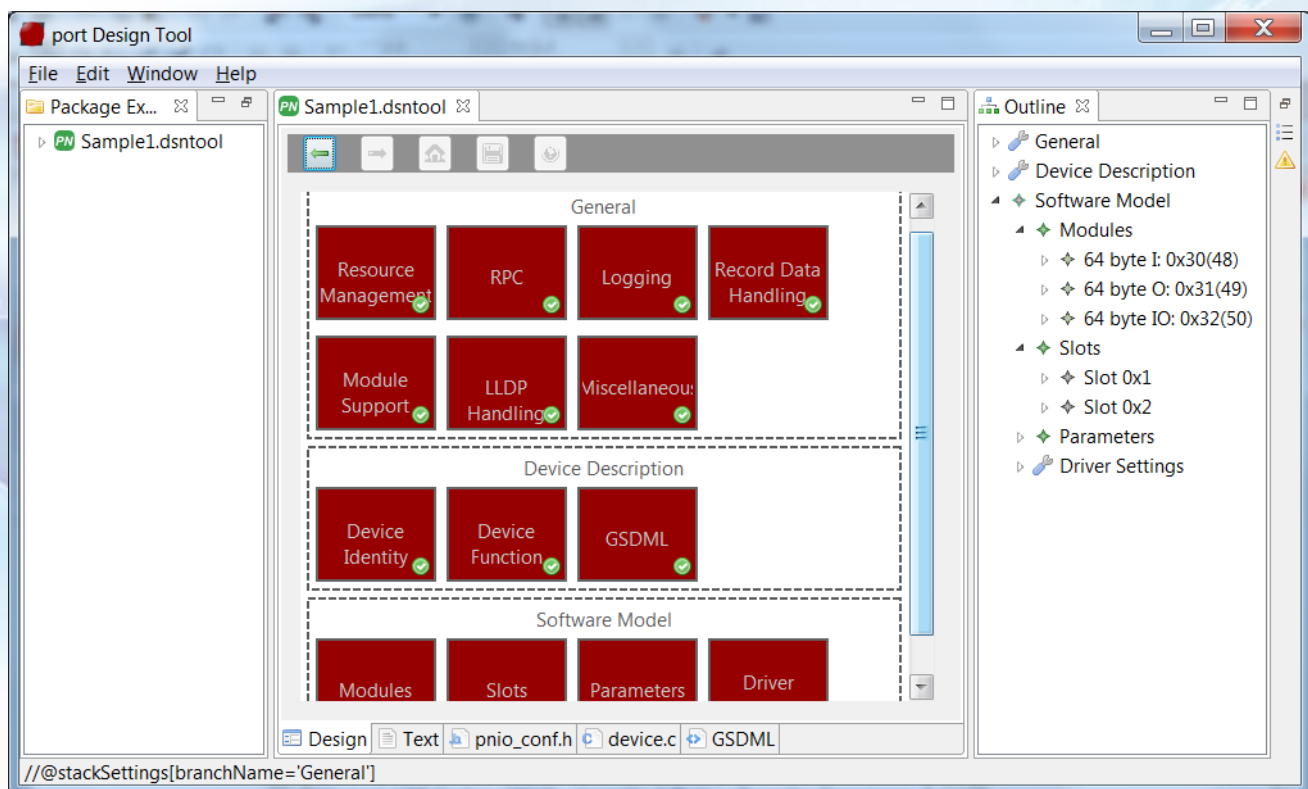
Complete ANSI-C Source Code with Example, Doxygen Manual, Reference Manual.

The Protocol Library specific PROFINET DesignTool significantly simplifies the engineering activities, creating a PROFINET device. Using the DesignTool, the engineer defines significant settings:

- Configuration of the engineered device
- Parameter of the PROFINET Protocol Library
- Communication Objects and default Parameters
- many more

The DesignTool generates accordingly:

- Configuration Files for the Library
- Variables Definitions for the customer's application
- Standard compliant GSDML-File



Besides significantly improving the engineer's activities and reducing the spent hour in development, the DesignTool enables for a reproducible, deterministic and storable development process.

The bundle PROFINET Protocol Library and PROFINET DesignTool does not only improve the **Time-to-Market**, it enables as well for maintaining a high quality in the development process since an automated and always identical reproducible configuration process is provided.

PN \*Sample1.dsntool

Save auto-save C:/tmp/PNDT/GSDML-V2.3-portExamp Browse

**Submodule ID=0\_0: 64 byte I**

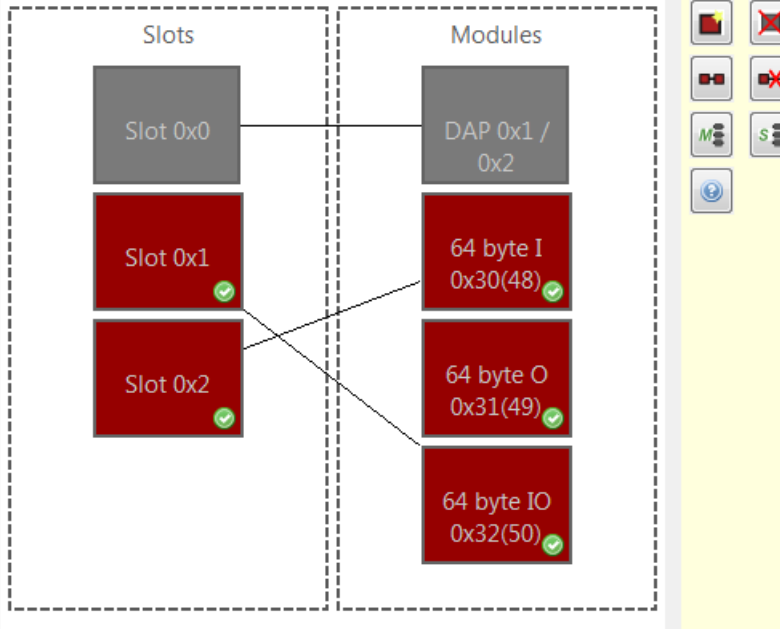
Submodule Ident Number	0x1
Information	
Assigned to API	0
Maximum i-Parameter Size	0

**Cyclic Input Data** All items consistency

Name	Data Type	Display as Bits	Length [Bytes]
PN_DATATYPE_OCTETSTR	OctetString	Yes	64
<b>pm_test_U16 (Index: 1 -- Length: 2 Byte -- Transfer Sequence: 0 -- Changeable With Bump: No )</b>			
<b>pm_test_U08 (Index: 22 -- Length: 1 Byte -- Transfer Sequence: 0 -- Changeable With Bump: No )</b>			
<b>pm_test_U32 (Index: 33 -- Length: 4 Byte -- Transfer Sequence: 0 -- Changeable With Bump: No )</b>			
<b>pm_test_146 (Index: 44 -- Length: 146 Byte -- Transfer Sequence: 0 -- Changeable With Bump: No )</b>			
<b>pm_test_U08 (Index: 45040 -- Length: 1 Byte -- Transfer Sequence: 0 -- Changeable With Bump: No )</b>			

Design Text pnio\_conf.h device.c GSDML

PN \*Sample1.dsntool



**Slots**

- Slot 0x0
- Slot 0x1 ✓
- Slot 0x2 ✓

**Modules**

- DAP 0x1 / 0x2
- 64 byte I 0x30(48) ✓
- 64 byte O 0x31(49) ✓
- 64 byte IO 0x32(50) ✓

Design Text pnio\_conf.h device.c GSDML