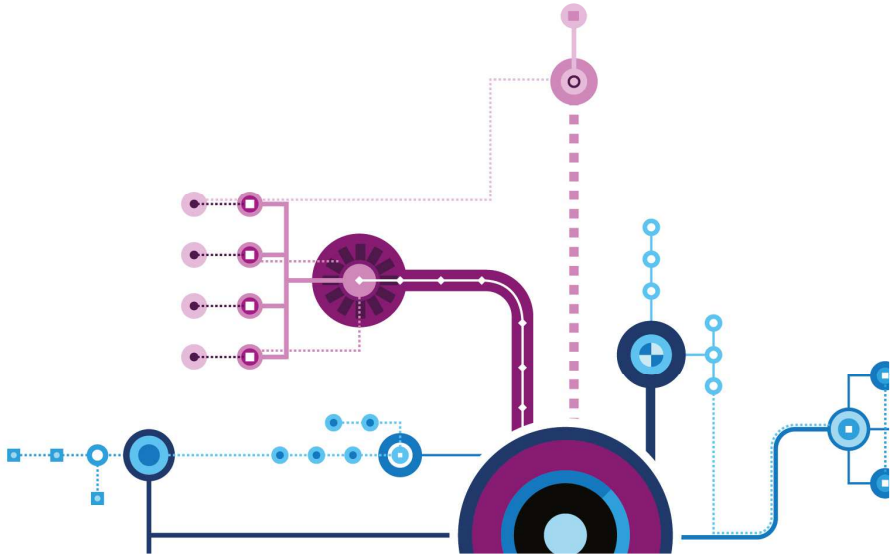
IBM Symposium Systèmes 2014

Concevoir plus rapidement des systèmes de plus en plus flexibles et complexes

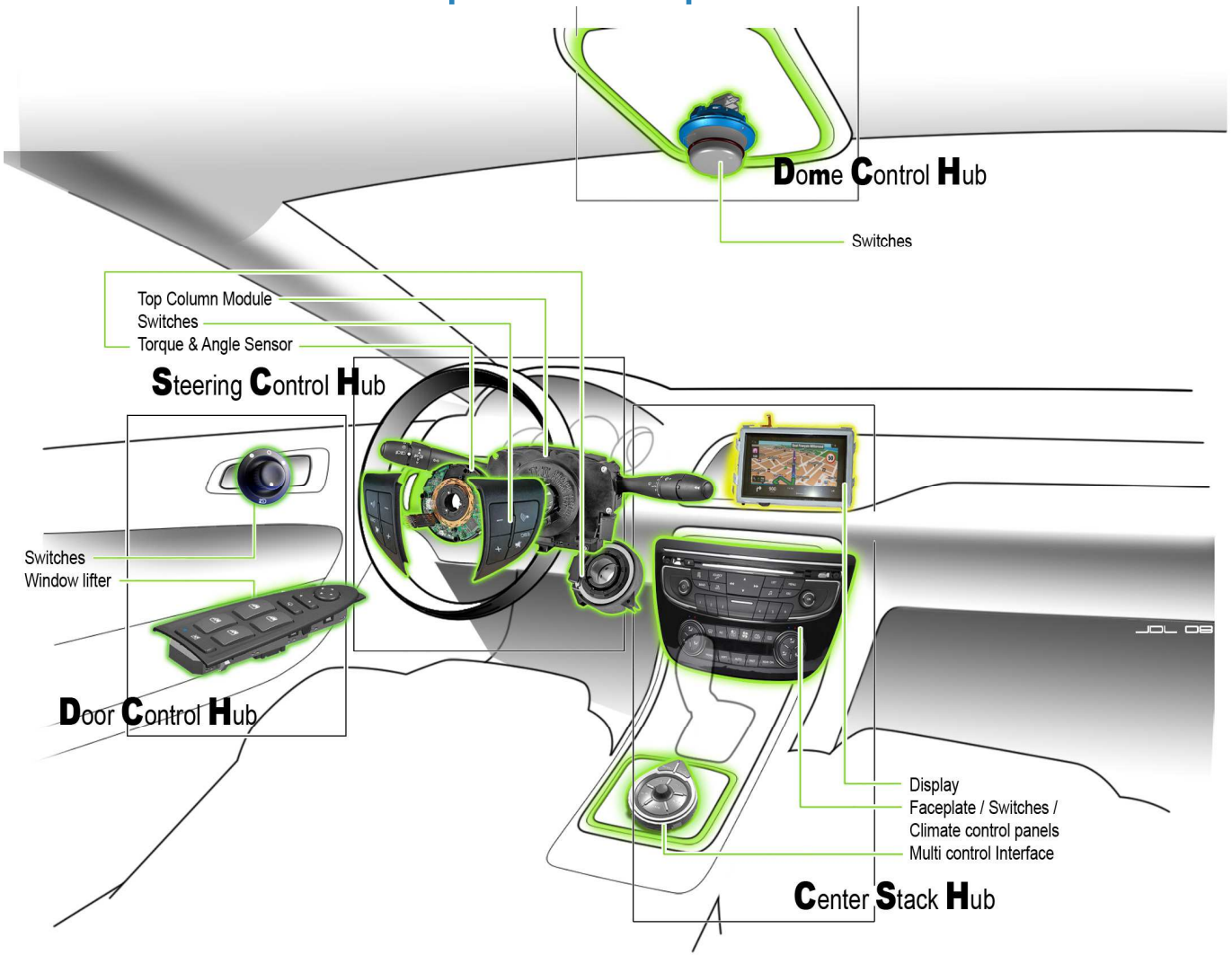
Rhapsody to generate µC low level code

Valeo

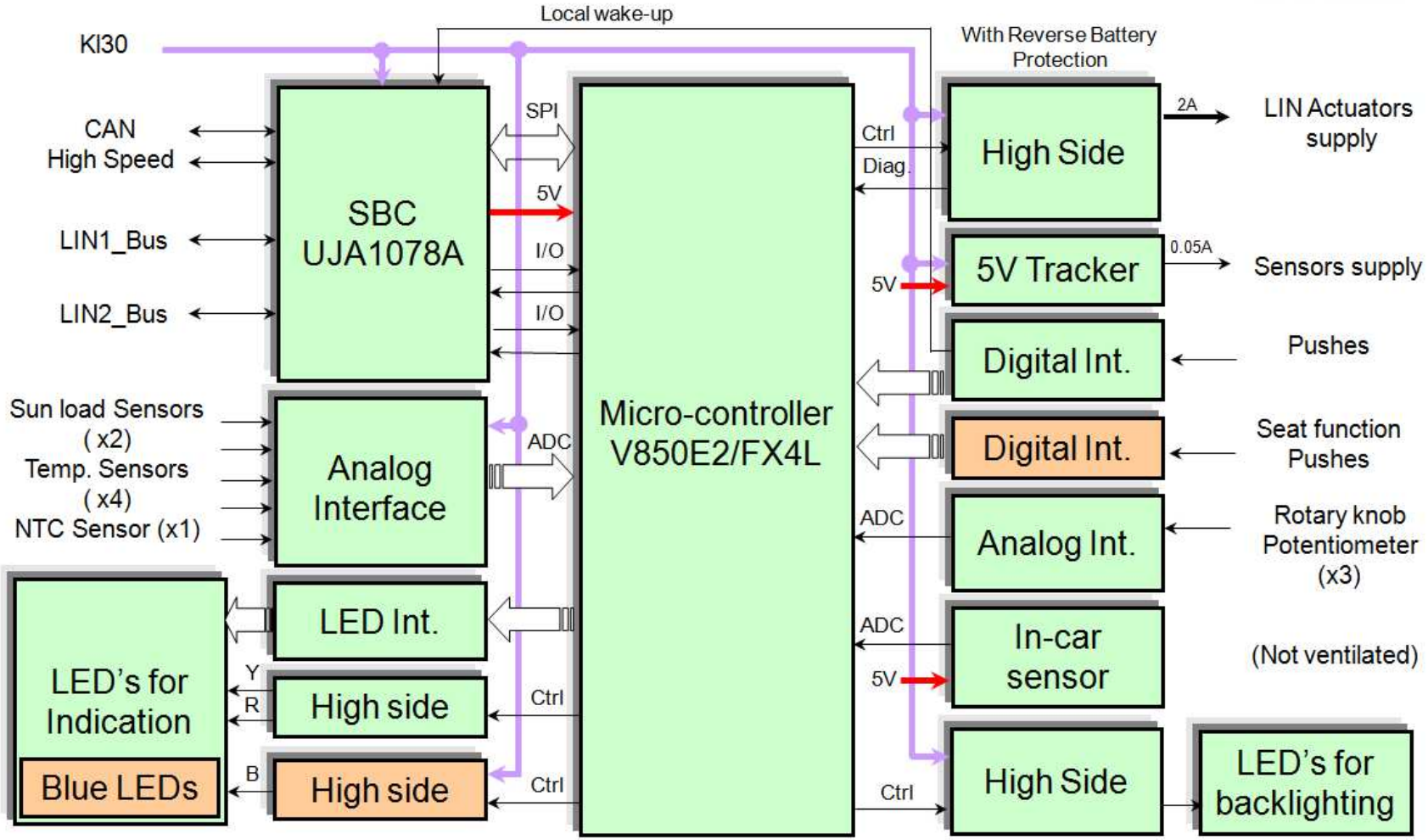
Jeudi 27 mars 2014
à l'IBM Client Center Paris



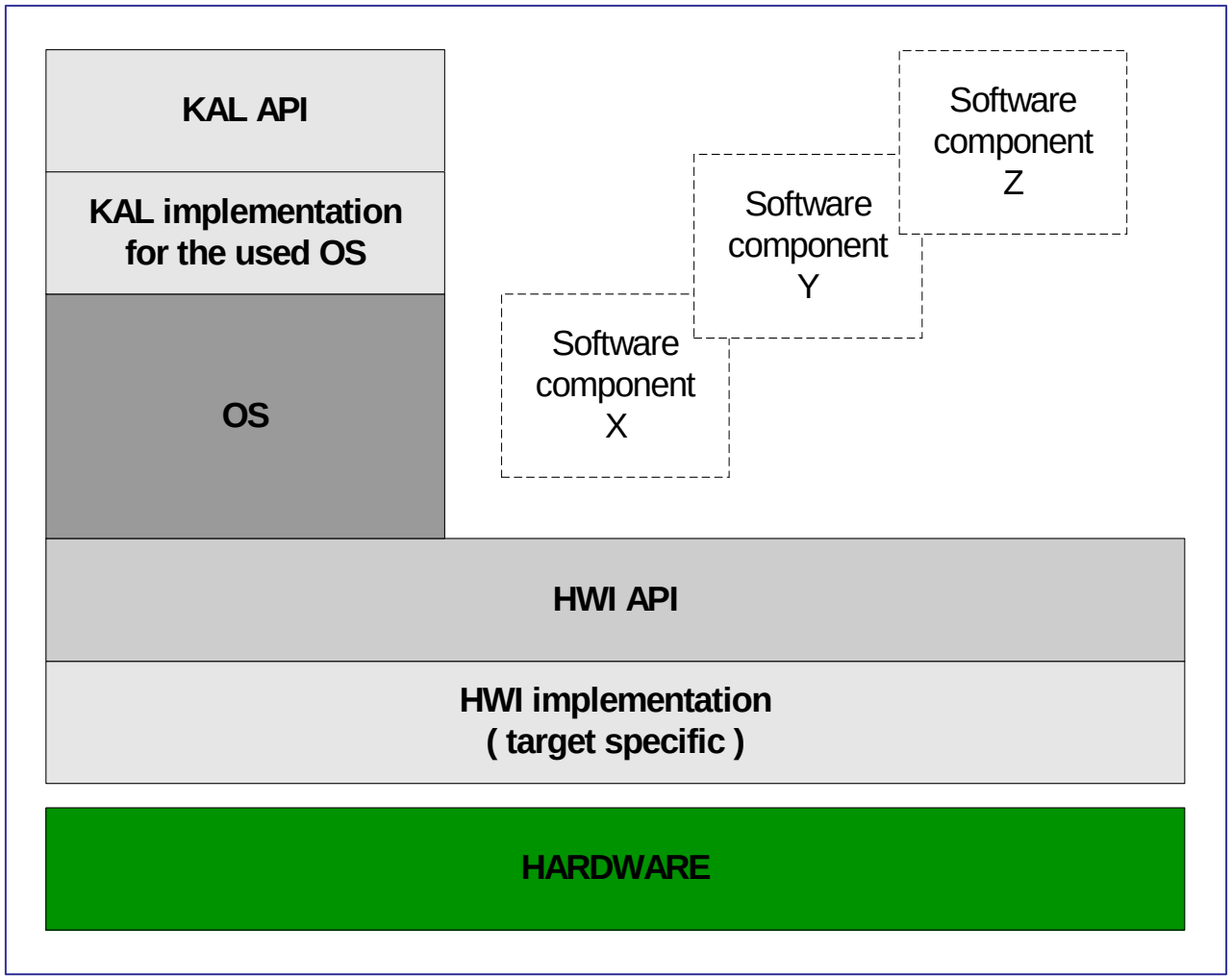
VALEO Interior Control products portfolio



Typical product architecture



A standard software architecture for all products



HWI is an hardware abstraction layer

- Standardized interface to access the hardware
- For a collection of objects with similar associate services (control, read, write, ...)
  - Digital inputs / outputs
  - Analogical inputs / outputs (e.g ADC)
  - Frequential inputs / outputs (e.g. PWM)
  - …

- Depending on hardware design, an implementation is made by each project without reusing code (no real standard, means tested, off the shelf)

HWI typical implementation : hand made

```
void FAR_FUNCTION HWI_vidStart(void)
{
  /* Ports Initialisation */
  /* P32 on TOAA01 */
  PMC3L  |= BIT_2;  /* Alternative mode */
  PFCE3L &= ~BIT_2; /* Function 1..2 */
  PFC3L  |= BIT_2;  /* Function 2 */

  /* P40 on SIB0 (CSIB) */
  PMC4  |= BIT_0;  /* Alternative mode */
  PFCE4 &= ~BIT_0; /* Function 1..2 */
  PFC4  &= ~BIT_0; /* Function 1 */

  /* P41 on SOB0 (CSIB) */
  PMC4  |= BIT_1;  /* Alternative mode */
  PFCE4 &= ~BIT_1; /* Function 1..2 */
  PFC4  &= ~BIT_1; /* Function 1 */

  /* P42 on SCKB0 (CSIB) */
  PMC4  |= BIT_2;  /* Alternative mode */
  PFCE4 &= ~BIT_2; /* Function 1..2 */
  PFC4  &= ~BIT_2; /* Function 1 */
```

An Opportunity for standardization :
Renesas (former NEC) µC product line at this time (2006)

- µC Series with same peripherals IP & several CPU (8 or 32 bits)
- In a family, a derivative is made by instantiating existing bricks (IP):
  - CPU : K0 or V850
  - Memories (ROM, RAM), several size
  - Peripherals (TAA, TAB, UARTD, …)

Table 1-1 V850ES/Fx3 features (2/2)

Series name		V850ES/FE3		V850ES/FF3		V850ES/FG3			
Product		'F3370A	'F3371	'F3372	'F3373	'F3374	'F3375	'F3376A	'F3377A
A/D Converter		10 bits x 10 ch		10 bits x 12 ch		10 bits x 16 ch			
Serial interfaces	UART/LIN	2 ch				3 ch		5 ch	
	CSI	2 ch							
	IIC	1 ch							
	CAN	1 ch				2 ch			
DMA		4 ch							
Interrupts	External (incl. NMI)	9 ch				12 ch		13 ch	
	Internal	48 ch				60 ch		65 ch	

Renesas µC product line : idea of a standard was born

- Peripherals are reused within a family
  - A software brick to manage a peripheral could be made
- Peripherals are instantiated n times for a derivative
  - The software brick should be instanciable (same code running several time with different set of data)

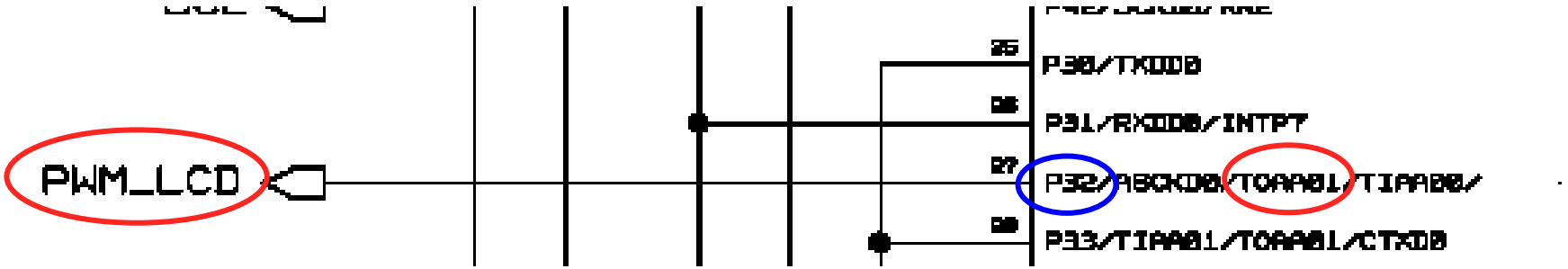
Commitment to design this standard with hand written code

- Result was an Heavy / difficult to understand software component
  - Rhapsody has been used to reverse / understand the code

With lot of parts already in a Rhapsody model, why not generating the code ?

Basic concepts of the standard : it should be easy to use !

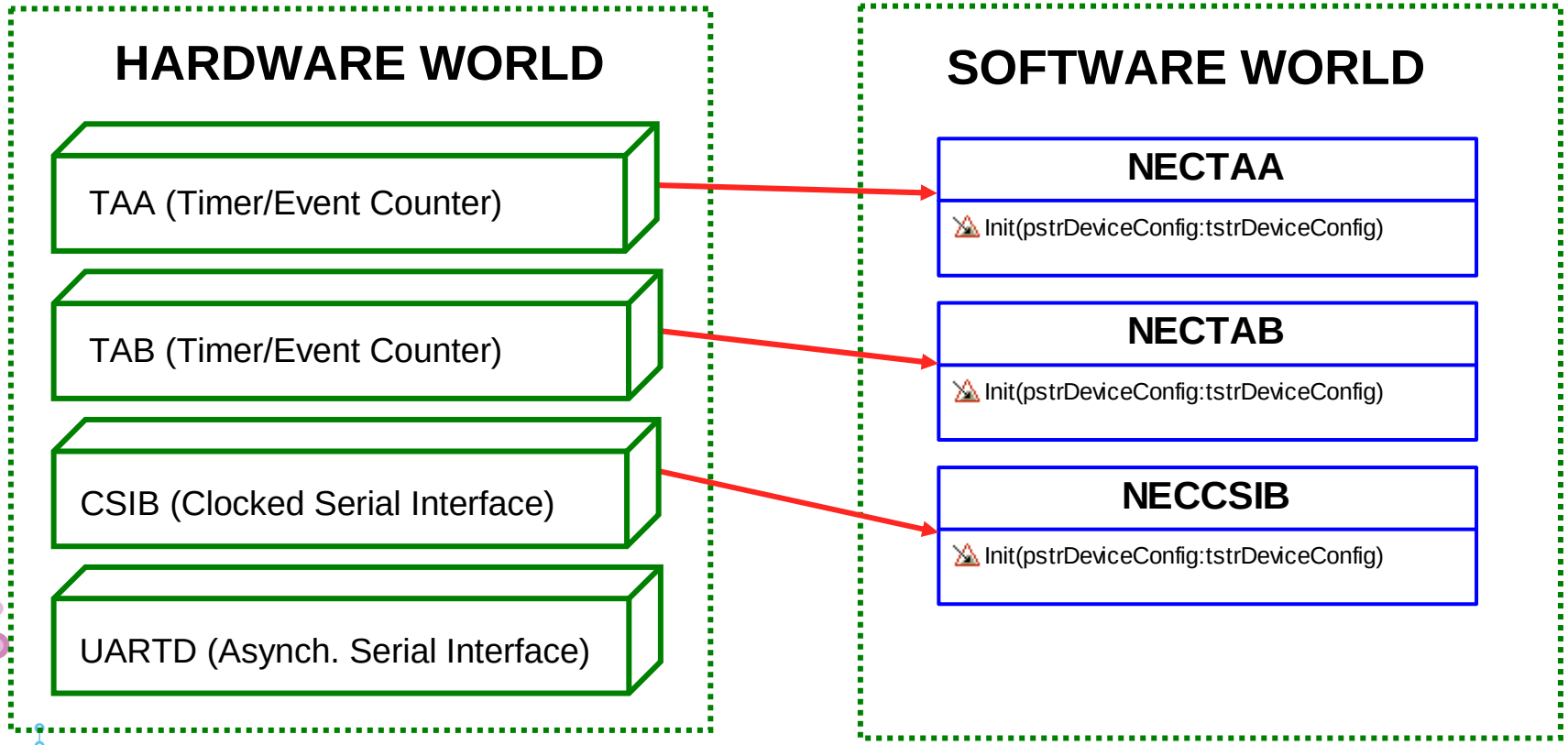
- Configuration driven by Hardware Schematics :
  - What is the usage of the pin in my project ?
    - I use it like this by writing the minimum of code (1 line)



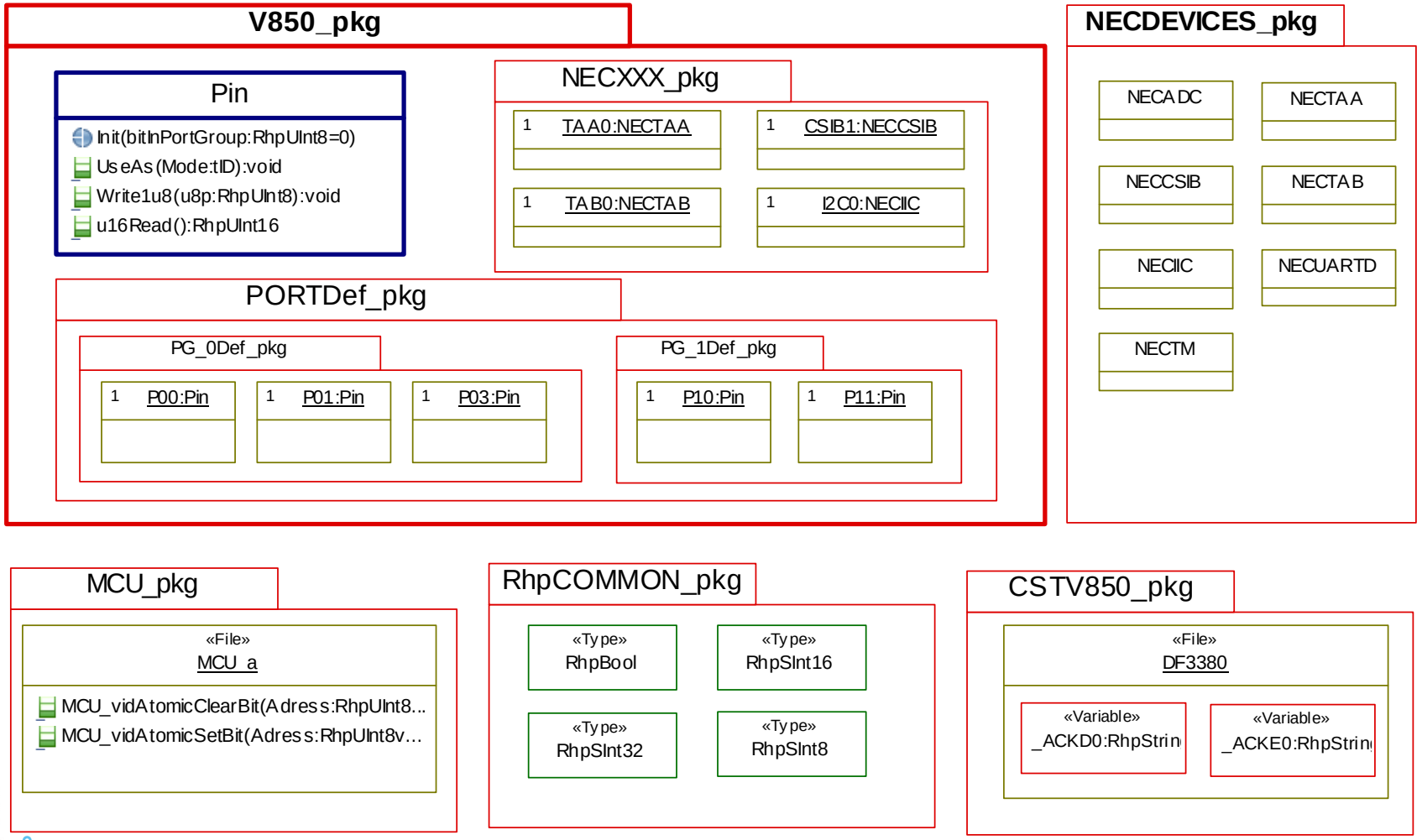

```
/* Use P32 as Frequential output */

Pin_UseAs(&P32, T0AAn1);
```

Basic concepts of the standard continued

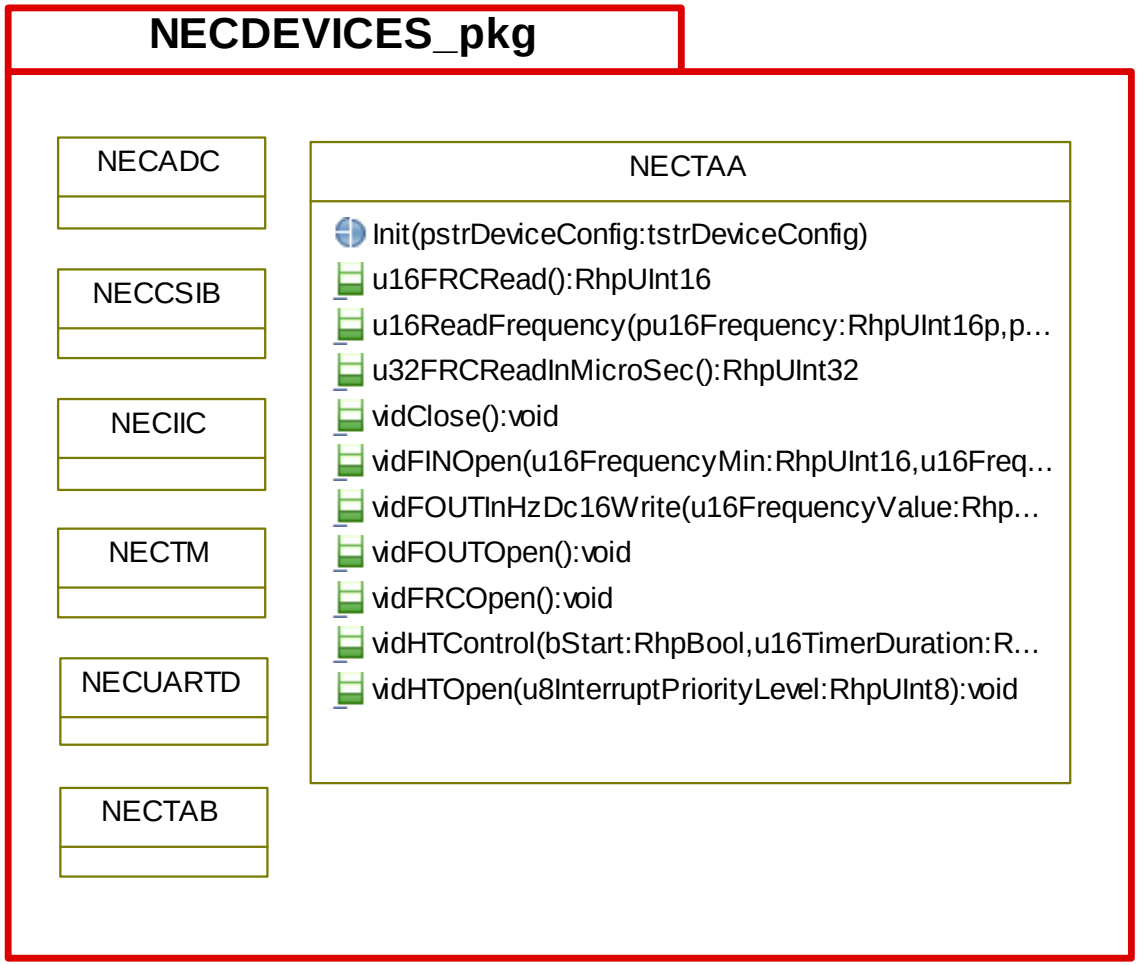
- Architecture based on physical objects



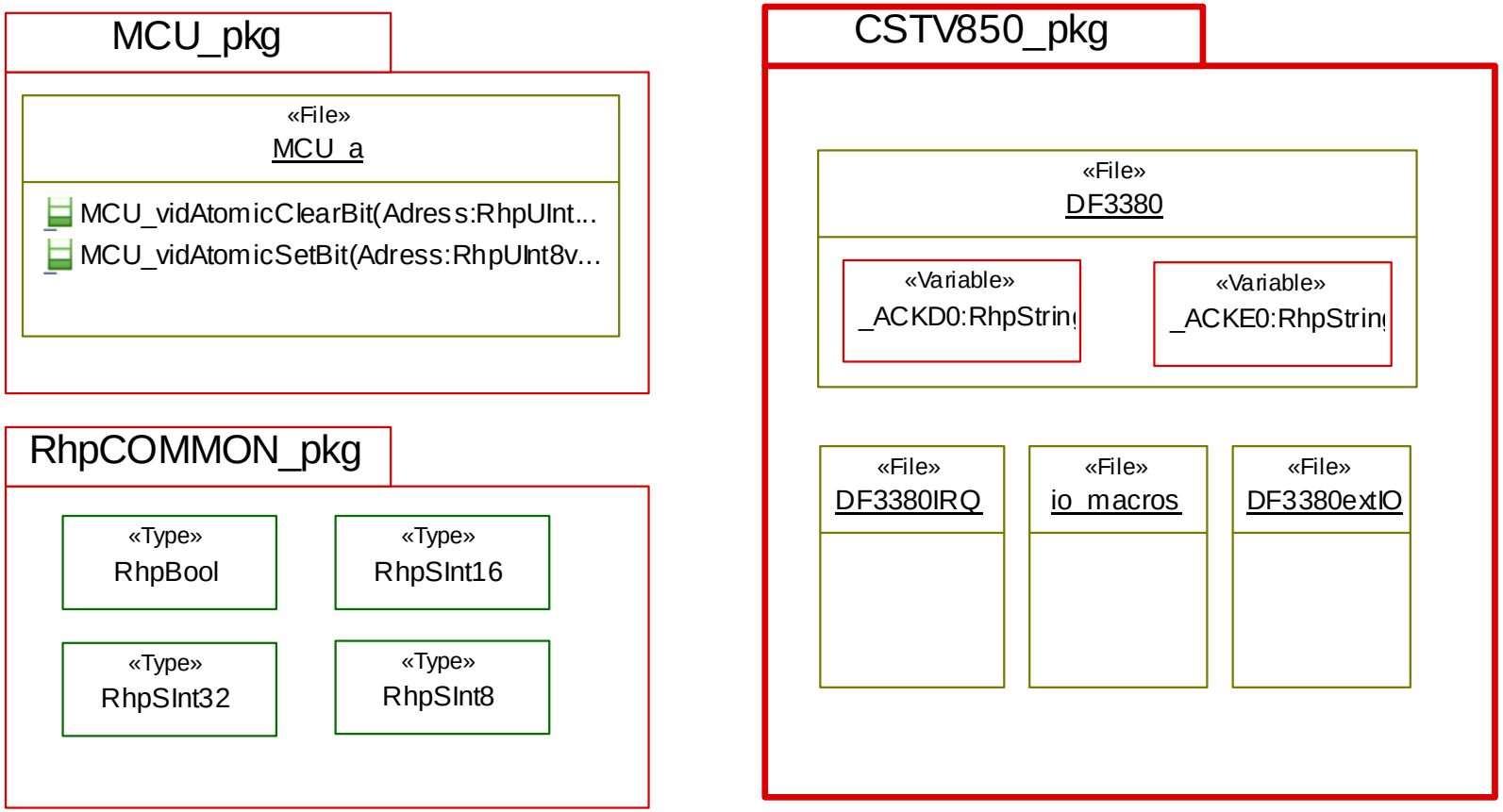
Rhapsody model – Global overview



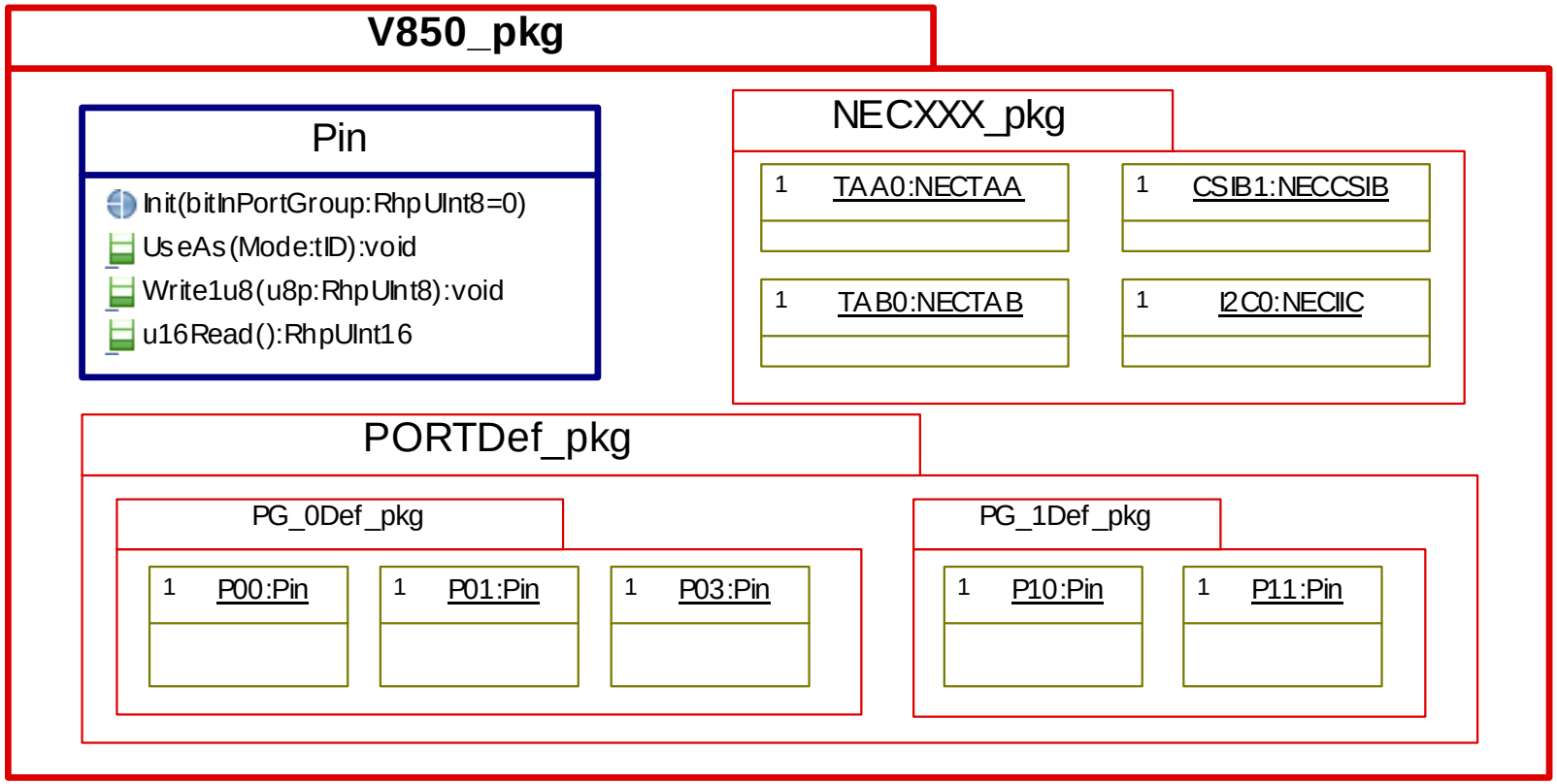
The NECDEVICES_pkg : classes for NEC Peripherals



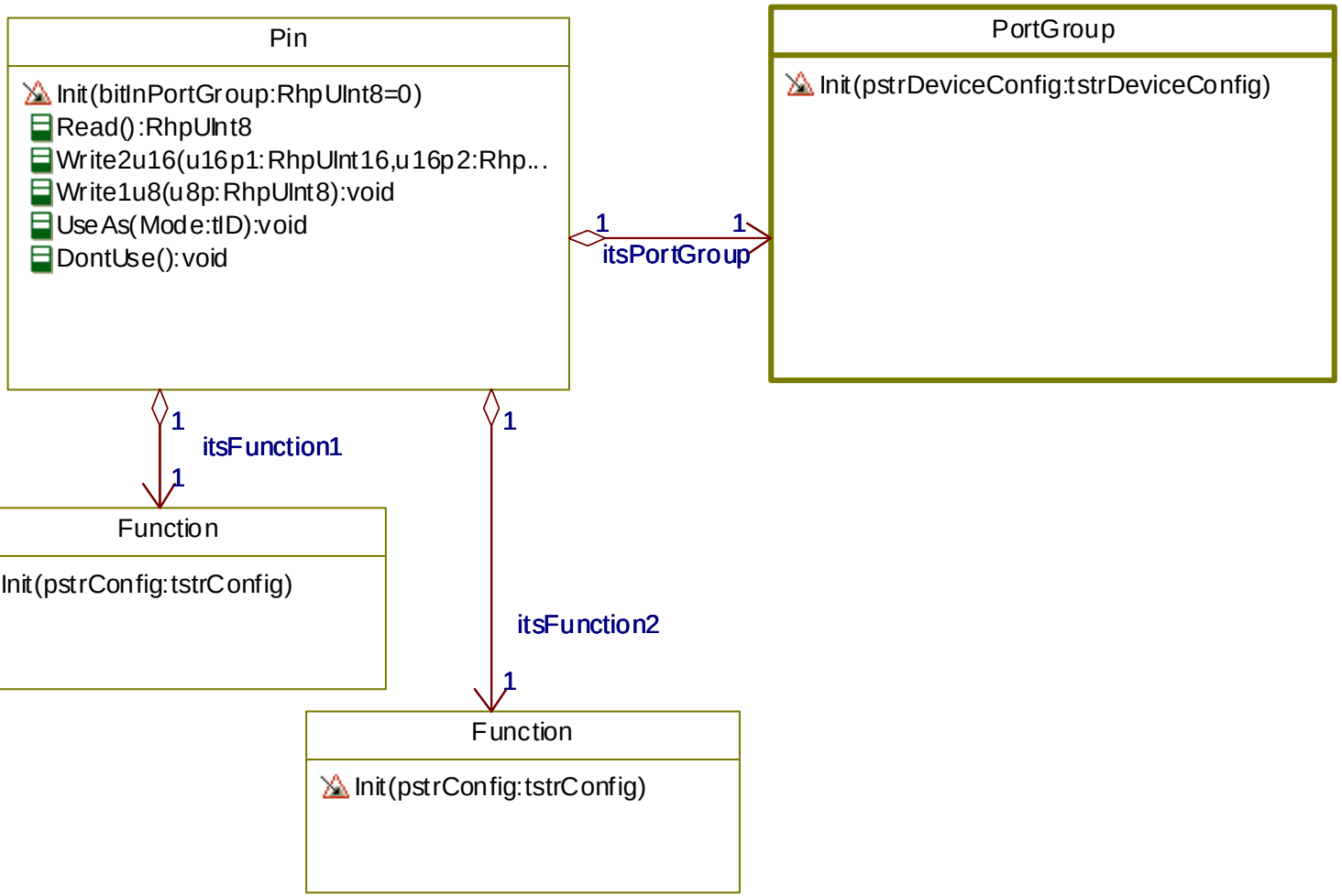
MCU_pkg, RhpCOMMON_pkg, CSTV850_pkg : facilities



V850_pkg : the µC related package



The Pin class at the heart of V850_pkg



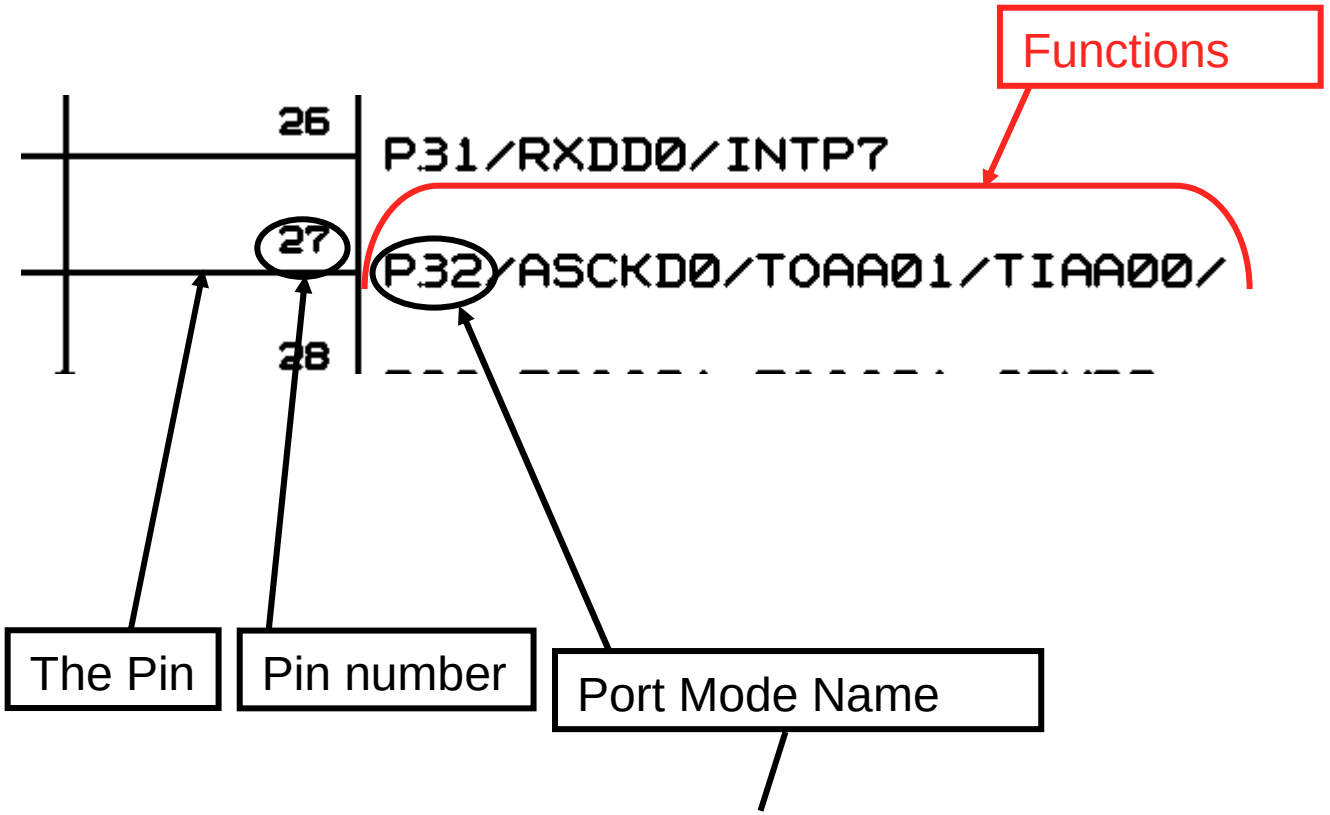
« Pin » , « Port Group » definitions from NEC

- Pin :
  - “Denotes the physical pin. Every pin is uniquely denoted by its pin number. A pin can be used in several modes. Depending on the selected mode, a pin name is allocated to the pin.”
- Port Group :
  - “Denotes a group of pins. The pins of a port group have a common set of port mode control registers.”

« Function » definition from NEC

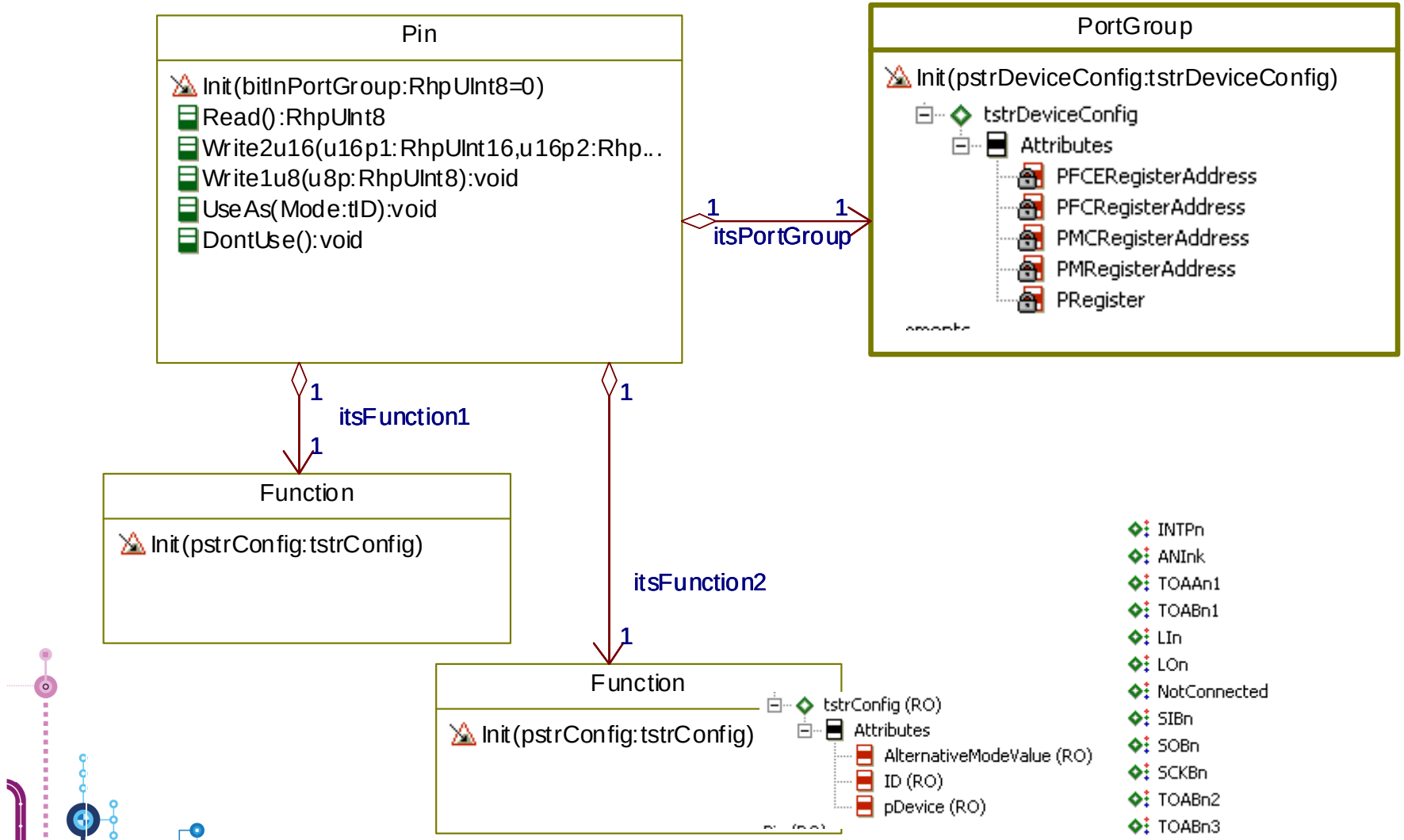
- Function
  - Port Mode
    - "A pin in port mode works as a general purpose input/output pin. It is then called "port". The corresponding name is Pnm. For example, P04 denotes port 4 of port group 0. It is referenced as "port P04"."
  - Alternative Mode
    - "In alternative mode, a pin can work in various non-general purpose input/output functions, for example, as the input/output pin of on-chip peripherals. The corresponding pin name depends on the selected function. For example, pin INTP0 denotes the pin for one of the external interrupt inputs. Note that for example P03 and INTP0 denote the same physical pin. The different names indicate the function in which the pin is being operated."

Pin and functions in the Hardware Schematics



used as a unique identifier because the pin number could be different depending of the µC derivative (FE, FG, FF, ...)

Pin, Port Group, Function in the Software World



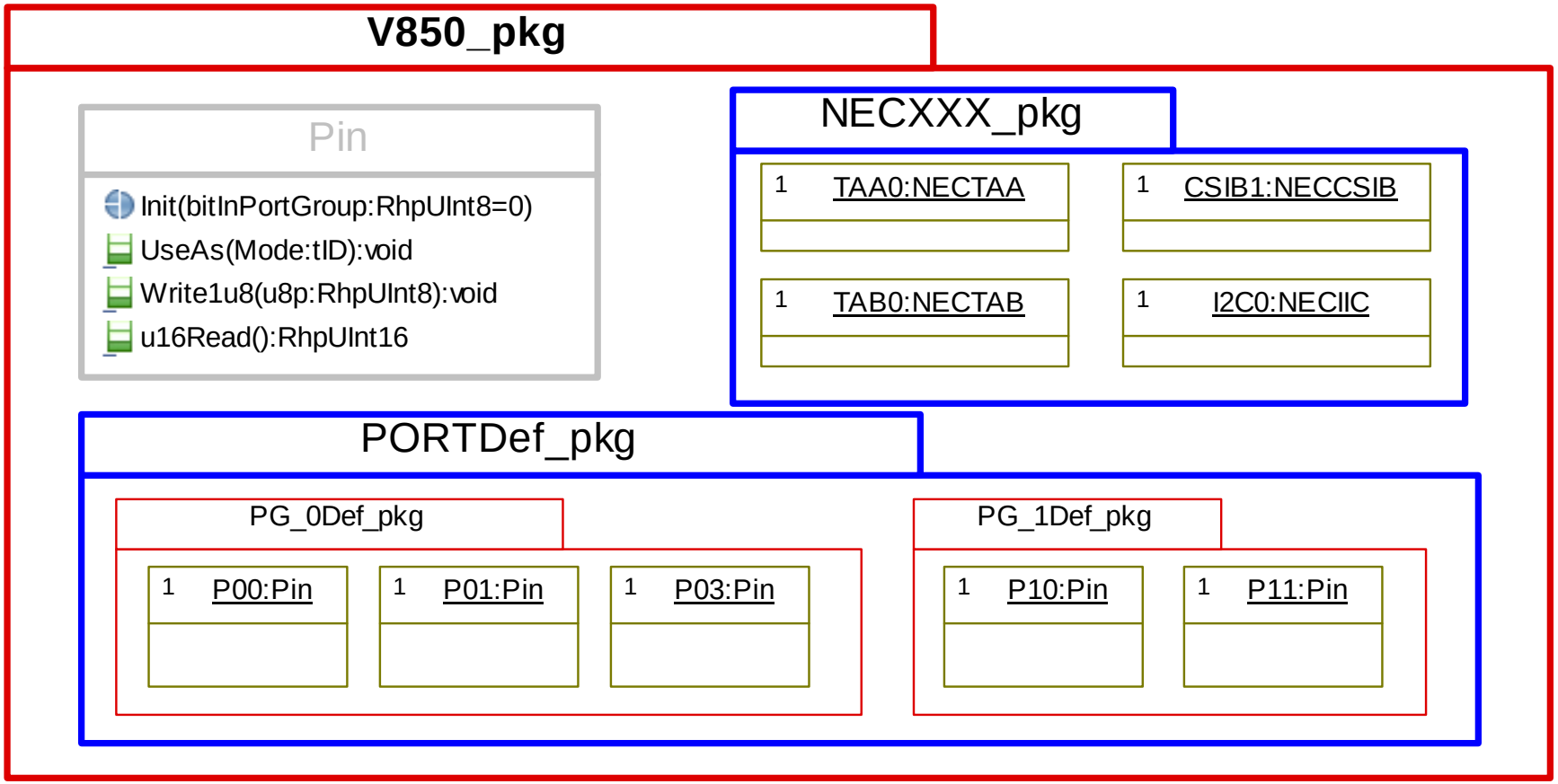
One line of code to use a pin, how it works ?

```
f = &ItsAvailableFunctions[0];
/* Search if the port has this function */
while ((!Found) && (*f != NULL))
{
  if ((*f)->pstrConfig->ID == Mode)
  {
    Found = TRUE;
    /* get port config for alternate mode from
    function object and apply it on registers */
    PFCE = (*f)->pstrConfig
    PFC  = (*f)->pstrConfig
  }
  else
  {
    f++;
```

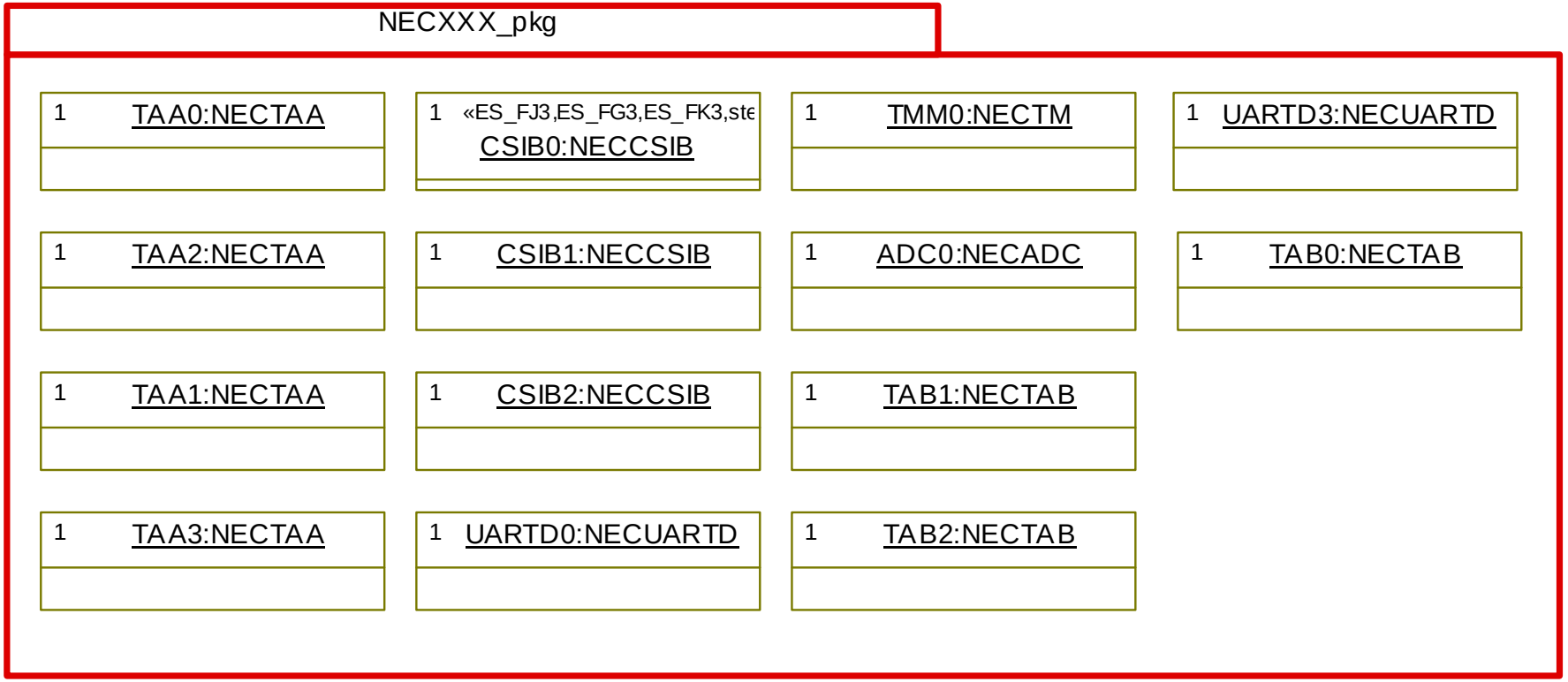
```
/* Use P32 as Frequential output */

Pin_UseAs(&P32, TOAAn1);
```

```
if (Found)
{
  /* Save used Mode and Device reference */
  me->mode = Mode;
  /* Get the related DeviceModule from function object */
  me->pDevice = (*f)->pstrConfig->pDevice;
  /* Configure DeviceModule regarding to function */
  switch(Mode)
  {
    case TOAAn1:
      /* Shall open the TAA associated in FOUT mode */
      NECTAA_vidFOUTOpen((NECTAA*)me->pDevice);
```

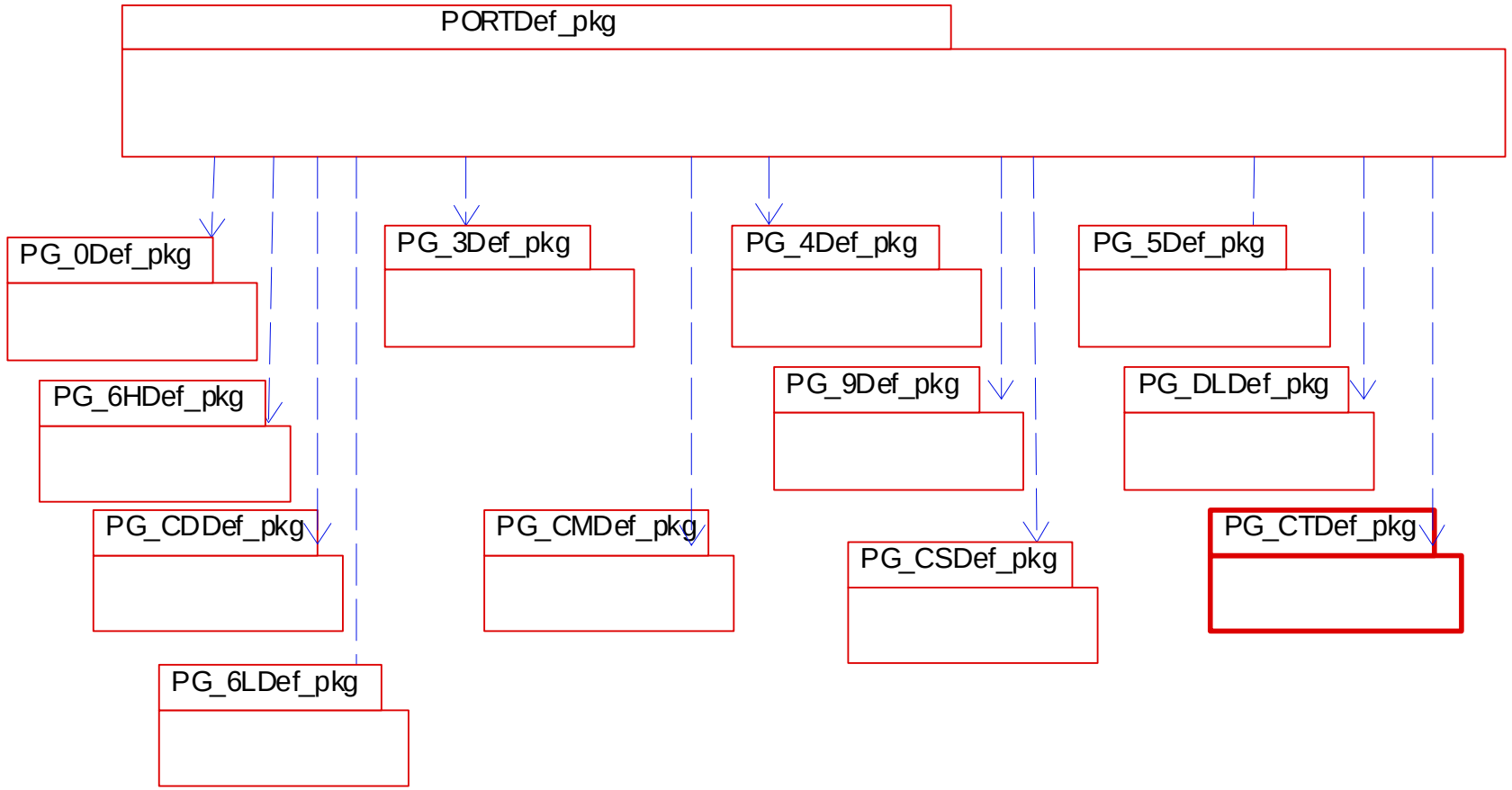
V850_pkg : **PORTDef_pkg & NECXX_pkg**



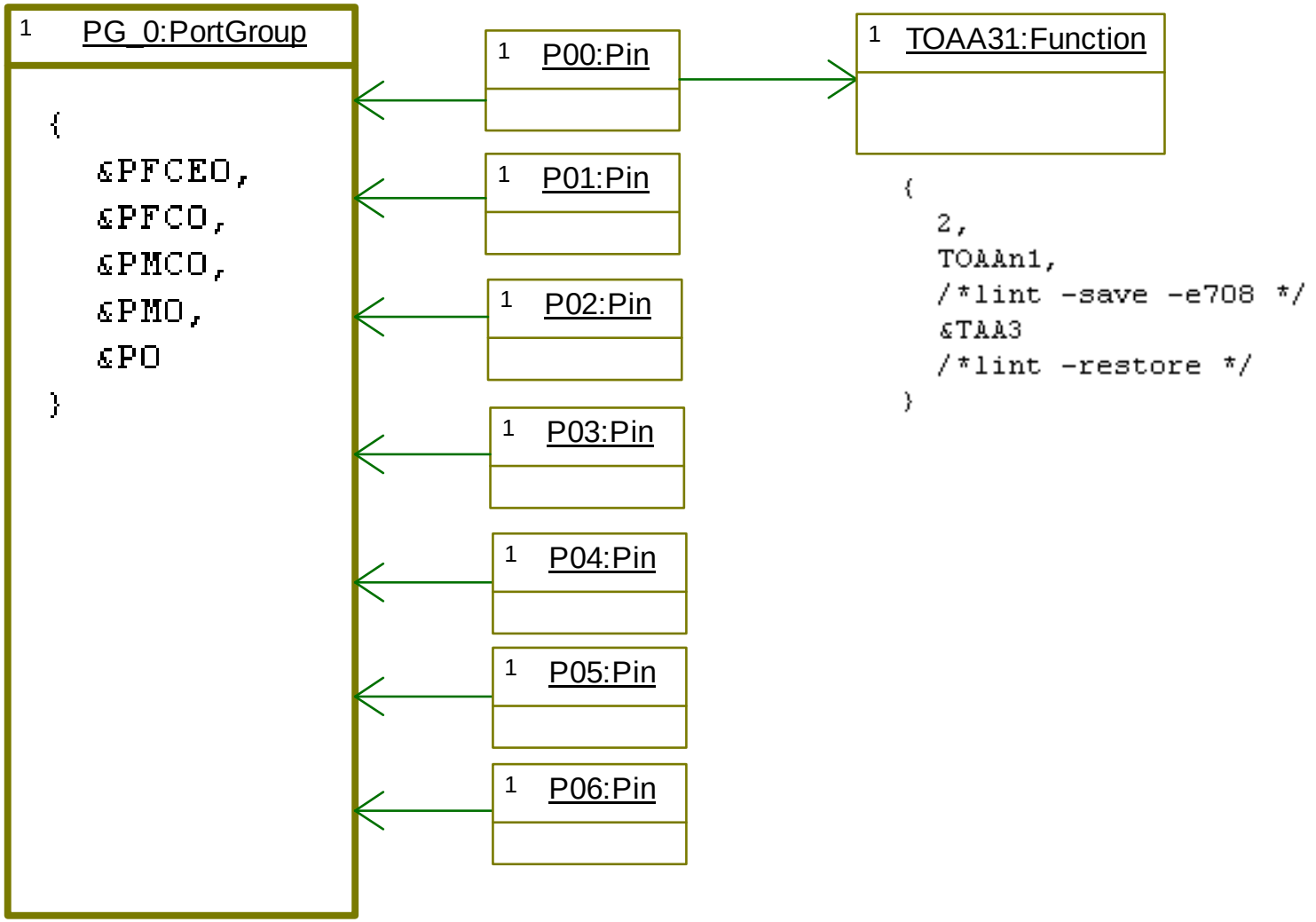
The NECXXX Package : instanciation of peripherals



PORTDef_pkg Package : Port groups objects



A Typical object model for one Port Group



Issues & solutions

- Huge usage of RAM memory
  - By default Rhapsody stores objects & relations in RAM
  - The solution was provided by the support team
    - A patch using properties to change the behavior of the code generator to have objects & relations in ROM memory
    - Now, integrated as a regular feature

- Overhead in code size : slower execution @ runtime
  - ROM is more & more bigger in µC : this is acceptable

- Long execution time at startup to configure the µC
  - A possible solution is to run the µC configuration off line and generate code associated to registers value writing

(e.g change the behavior of “MCU_pkg” to have C code to embed instead of writing registers)

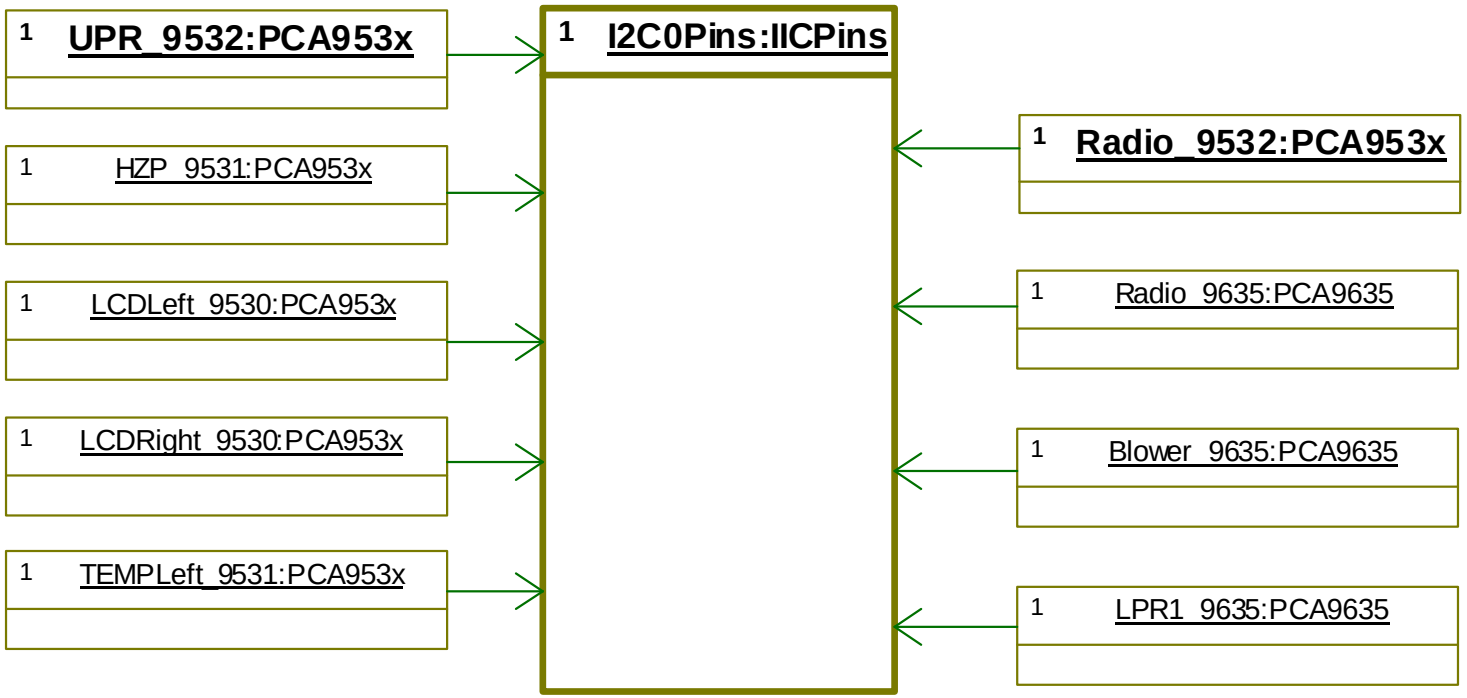
Benefits

- Cost reduction of HWI (Hardware abstraction layer) implementation
  - From 500h of work to ~100h

- Code is in the street : Peugeot, Mercedes, VW, Skoda…

Extension

- In innovation projects, extension has been made to manage IO expanders over I2C bus



IBM Symposium Systèmes 2014

Concevoir plus rapidement des systèmes de plus en plus flexibles et complexes

Thank you

Valeo

Jeudi 27 mars 2014
à l'IBM Client Center Paris