This guide explains how to disable the reset pin and use it as GPIO in Kinetis L devices:

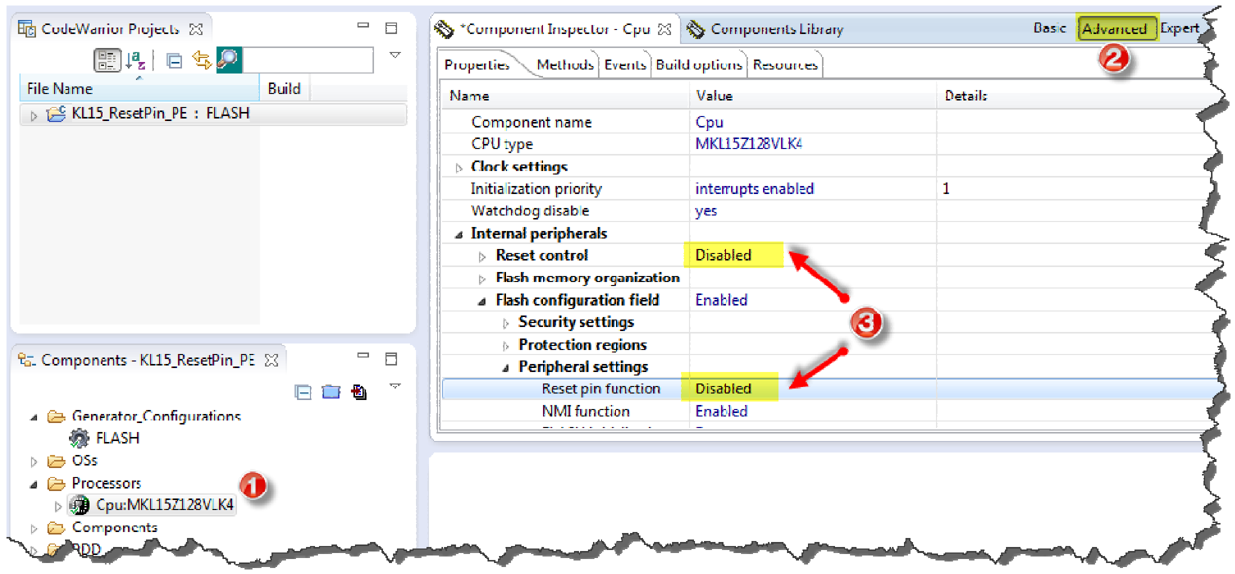
A) With Processor Expert
B) Without using Processor Expert

A) DISABLE RESET PIN AND USE IT AS GPIO WITH PROCESSOR EXPERT

1- Create a new project with processor expert support

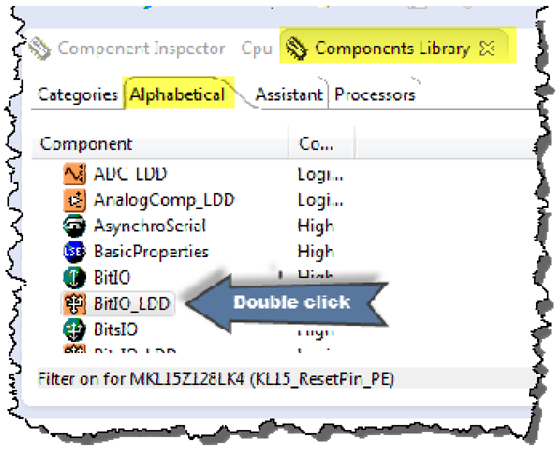
2- Double click on the CPU component, click on "Advanced" and modify the next properties:

- Internal peripherals -> Flash configuration field -> Peripheral settings -> Reset pin function (**Disabled**)
- Internal peripherals -> Reset control (**Disabled**)



Name	Value	Details
Component name	Cpu	
CPU type	MKL15Z128VLK4	
Clock settings		
Initialization priority	interrupts enabled	1
Watchdog disable	yes	
Internal peripherals		
Reset control	Disabled	
Flash memory organization		
Flash configuration field	Enabled	
Security settings		
Protection regions		
Peripheral settings		
Reset pin function	Disabled	
NMI function	Enabled	

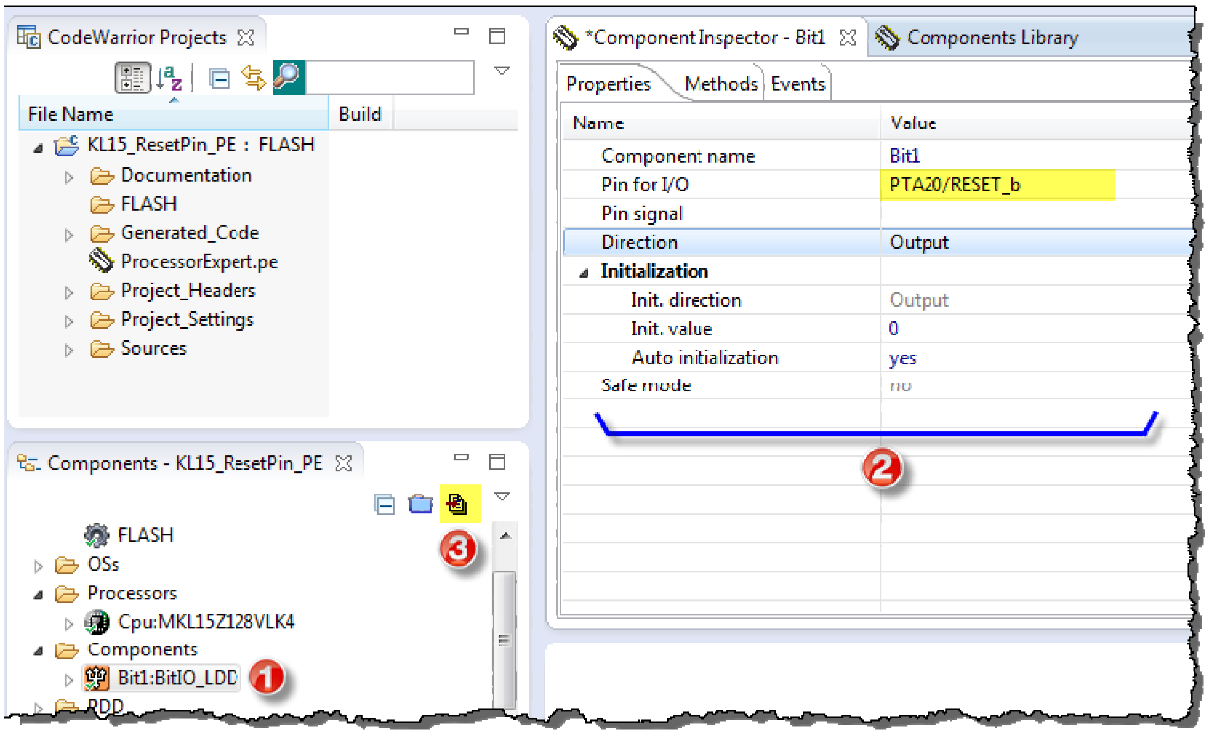
3- Go to Components library -> Alphabetical and add a 'BitIO_LDD' component with double click.



Component	Co...
ADC_LDD	Logi...
AnalogComp_LDD	Logi...
AsynchroSerial	High
BasicProperties	High
BitIO	High
BitIO_LDD	
BitsIO	High

Filter on for MKL15Z128LK4 (KL15_ResetPin_PE)

4- Configure it as needed (PTA20, input/output, auto-init, etc) from the component inspector. And finally click on the “Generate Processor Expert Code” icon.



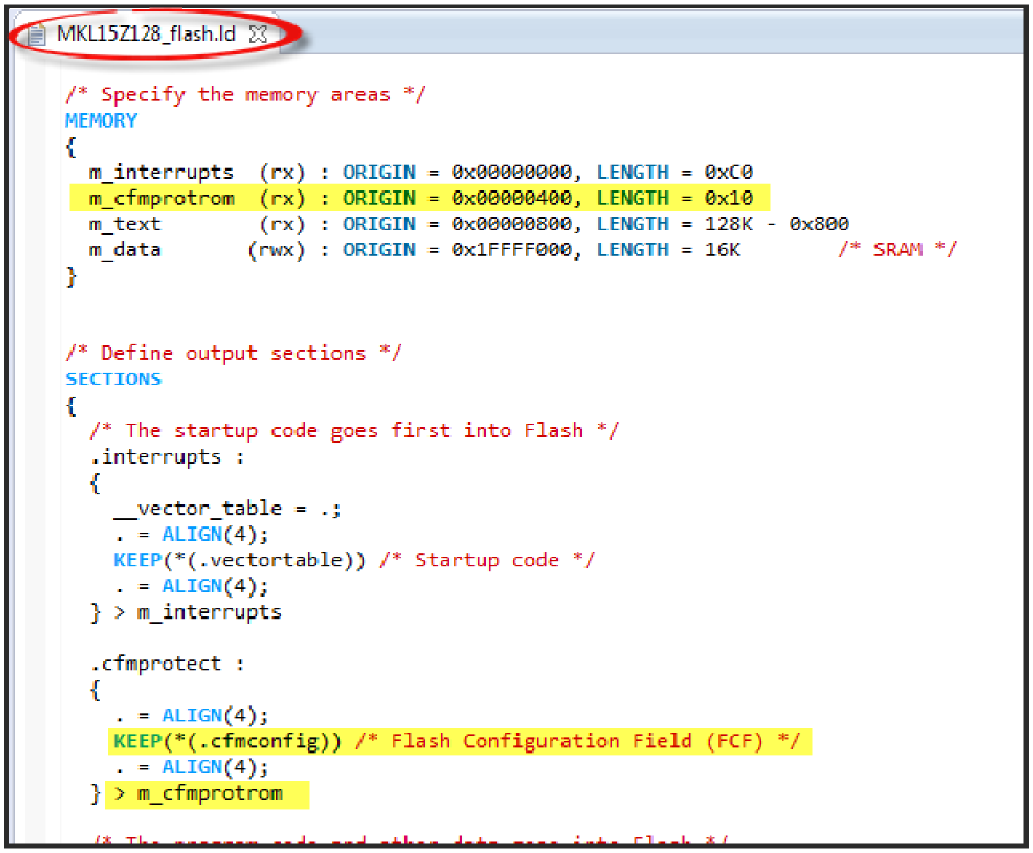
Name	Value
Component name	Bit1
Pin for I/O	PTA20/RESET_b
Pin signal	
Direction	Output
Initialization	
Init. direction	Output
Init. value	0
Auto initialization	yes
Safe mode	no

And the pin can now be used as GPIO with the methods provided by Processor Expert.

B) DISABLE RESET PIN AND USE IT AS GPIO WITH SIMPLE CODE

1- Create a new project WITHOUT processor expert.

2- From your project, go to the linker file (e.g. KL15Z128M4_flash.ld) and verify you have a "m_cfmprotrom" memory area (this MUST be at 0x00000400) and a '.cfmconfig' section placed in such area.



```
/* Specify the memory areas */
MEMORY
{
  m_interrupts  (rx) : ORIGIN = 0x00000000, LENGTH = 0xC0
  m_cfmprotrom  (rx) : ORIGIN = 0x00000400, LENGTH = 0x10
  m_text        (rx) : ORIGIN = 0x00000800, LENGTH = 128K - 0x800
  m_data       (rwx) : ORIGIN = 0x1FFFF000, LENGTH = 16K        /* SRAM */
}


/* Define output sections */
SECTIONS
{
  /* The startup code goes first into Flash */
  .interrupts :
  {
    __vector_table = .;
    . = ALIGN(4);
    KEEP(*(.vectortable)) /* Startup code */
    . = ALIGN(4);
  } > m_interrupts

  .cfmprotect :
  {
    . = ALIGN(4);
    KEEP(*(.cfmconfig)) /* Flash Configuration Field (FCF) */
    . = ALIGN(4);
  } > m_cfmprotrom
```

3- In one of the startup files (e.g. kinetis_sysinit.c) include the array definition of the next page (**be careful** not to change the values, specially the NV_FSEC, as this could brick the device). This structure is loaded to flash and contains the RESET_PIN_CFG bit set to 0, which disables the reset pin.

The user can now include instructions in code to configure the pin and use it as GPIO.

```
 /* Flash configuration field */
 __attribute__ ((section (".cfmconfig"))) const uint8_t _cfm[0x10] = {
  /* NV_BACKKEY3: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY2: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY1: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY0: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY7: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY6: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY5: KEY=0xFF */
   0xFFU,
  /* NV_BACKKEY4: KEY=0xFF */
   0xFFU,
  /* NV_FPROT3: PROT=0xFF */
   0xFFU,
  /* NV_FPROT2: PROT=0xFF */
   0xFFU,
  /* NV_FPROT1: PROT=0xFF */
   0xFFU,
  /* NV_FPROT0: PROT=0xFF */
   0xFFU,
  /* NV_FSEC: KEYEN=1,MEEN=3,FSLACC=3,SEC=2 */
   0x7EU,
  /* NV_FOPT:
??=1,??=1,FAST_INIT=1,LPBOOT1=1,RESET_PIN_CFG=0,NMI_DIS=1,??=1
,LPBOOT0=1 */
   0xF7U,
   0xFFU,
   0xFFU
 };
```