CodeWarriors & Processor Expert Seminar

AMF-GEN-T0011

CodeWarriors IDE and Processor Expert In-depth Training
Presented by Hy Mai and Ruth Rhoades

- How to position and sales Processor Expert
- LAB1 -- Blinking LED (Timer and I/O)
- LAB2 – KBI & Analog Compartor
- LAB3 -- ADC and PWM
- LAB4 – SCI (Uart) Communication with PC
- LAB5 – SPI Serial Peripheral Comm neighbor's board
- LAB6 – I2C Comm with neighbor's board
- LAB7 – Input Capture to measure Pulsewidth (Optional)
- LAB8 – Make check payable to He My FAE

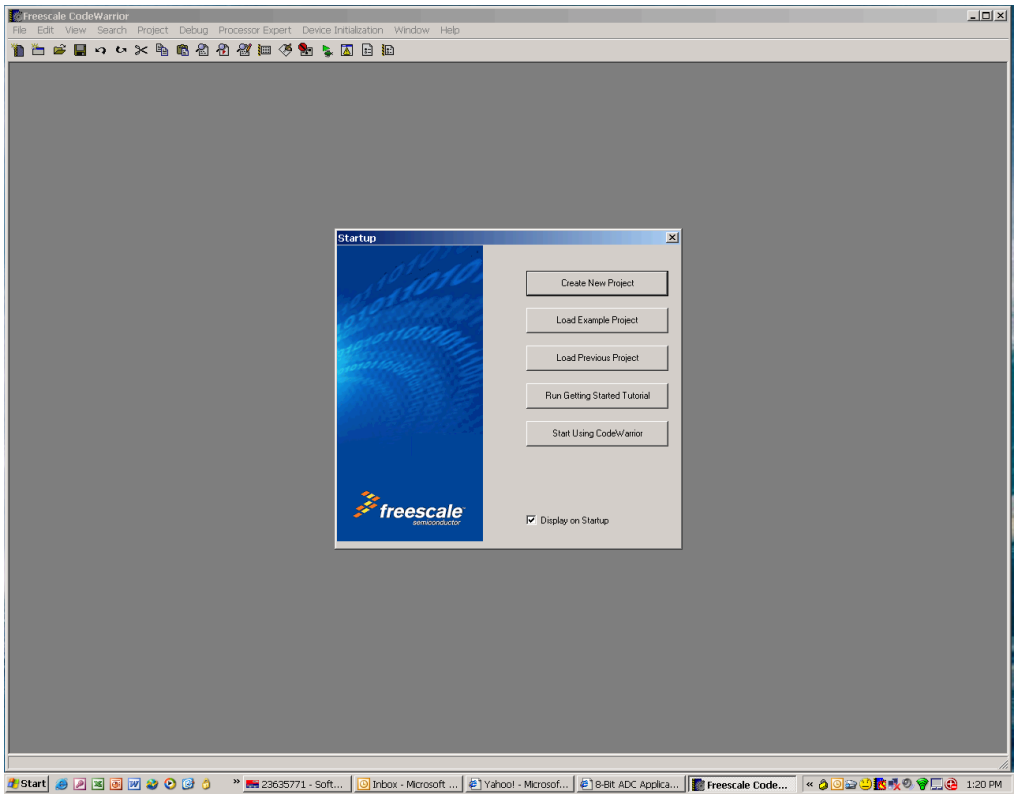
freescale semiconductor

LAB1 -- Blinking LED

- Configure Bus Clock to 8Mhz
- Set Timer to generate an interrupt every 250ms
- Configure output on LED2 (PORTB7)
- Toggle LED at every Timer Interrupt

LAB1 -- Blinking LED

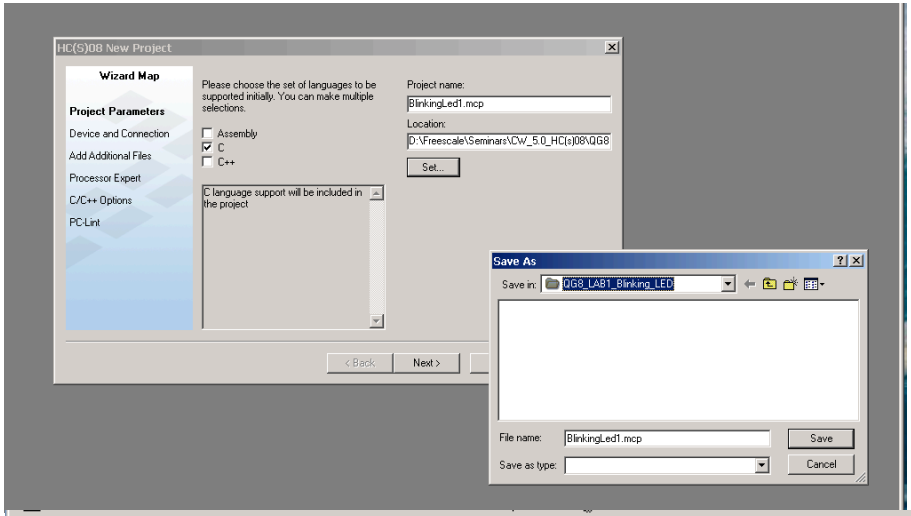
- Click “Create New Project”



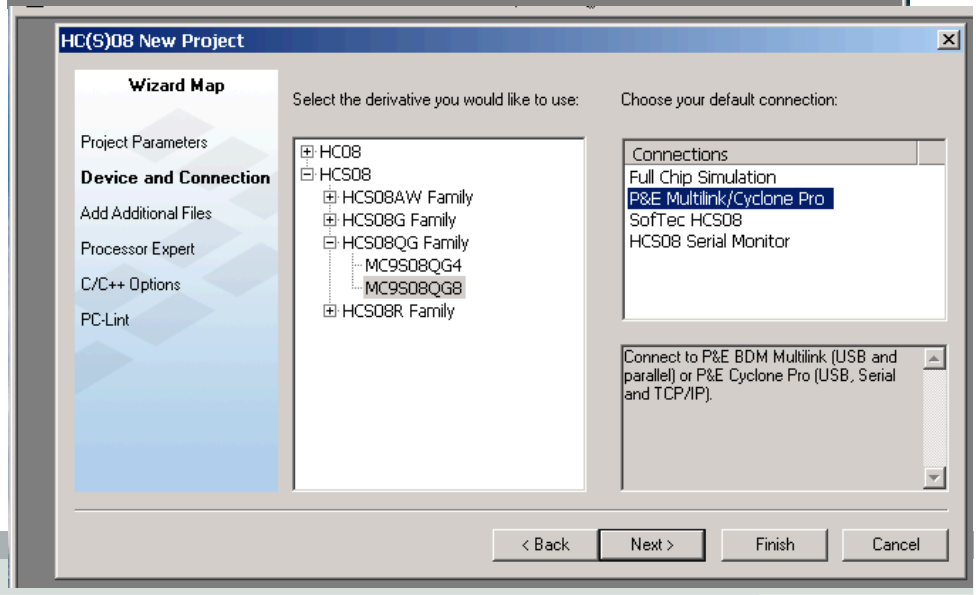
freescale™ semiconductor

LAB1 -- Blinking LED

►Click on the “Set” button to set the directory
►Give the project a name and hit “Save” and then hit “Next”



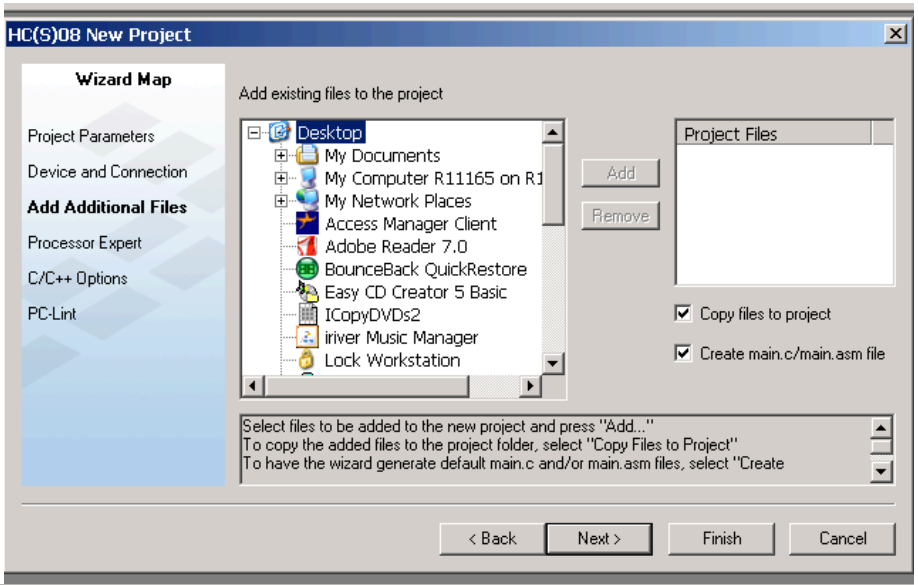
•Expand the “HCS08” directory structure to and find the “MC9S08QG8” device
•Select “P&E Multilink/Cyclone Pro” under the Connections box, then hit “Next”



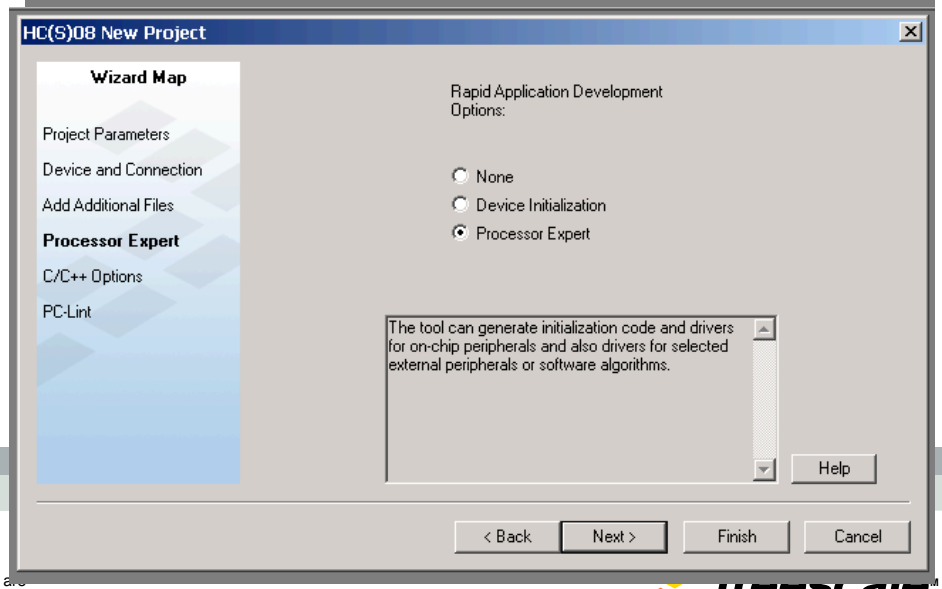
freescale semiconductor

LAB1 -- Blinking LED

►Since we are not importing any existing files just hit “Next”



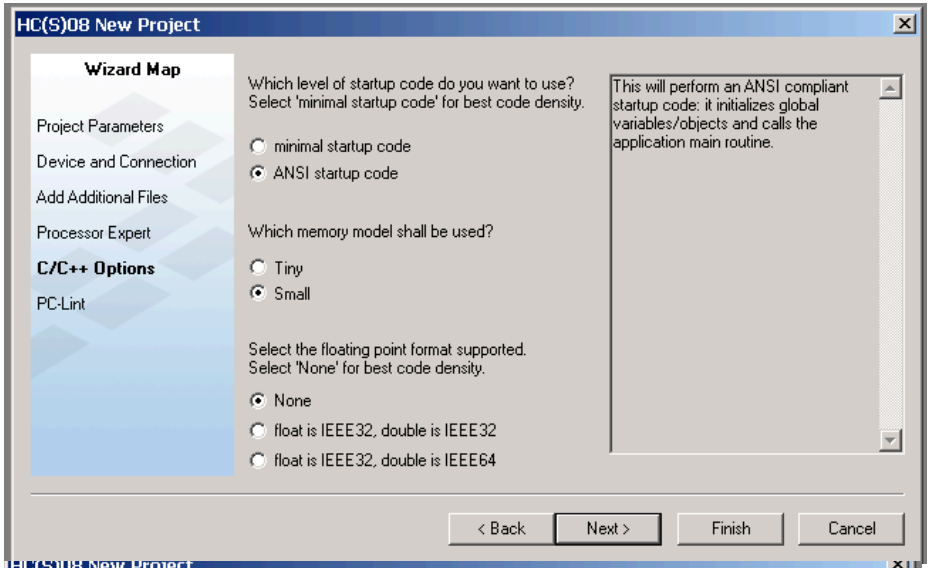
►Select “Processor Expert” and hit “Next”



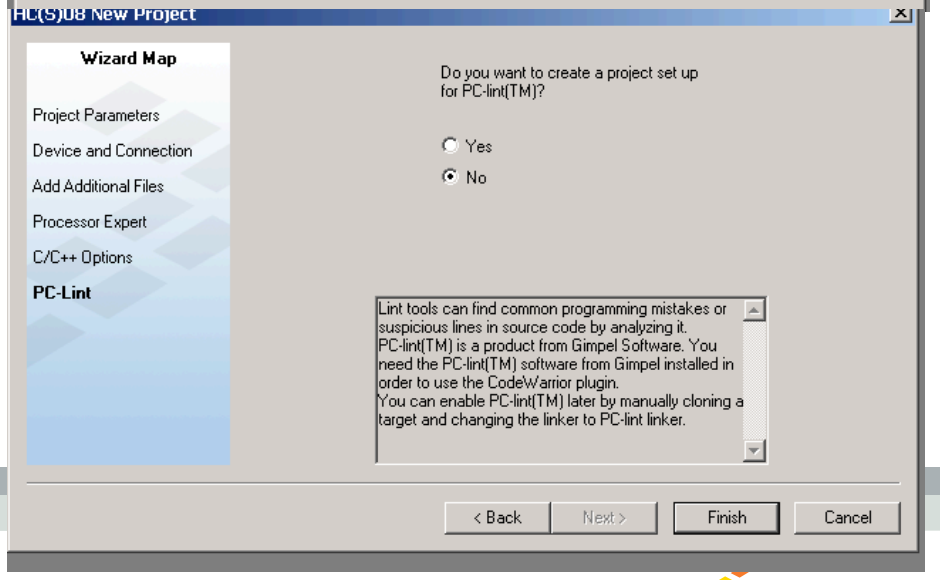
freescale semiconductor

LAB1 -- Blinking LED

►Leave everything as default and click "Next"



►Leave everything as default and click "Finish"

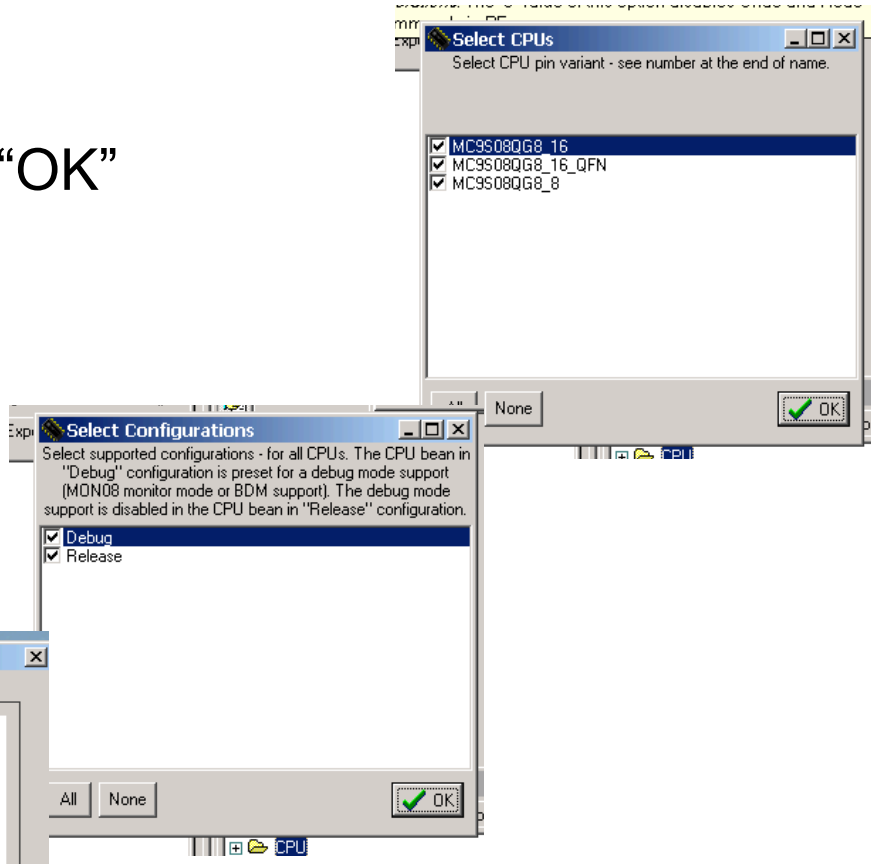


freescale semiconductor

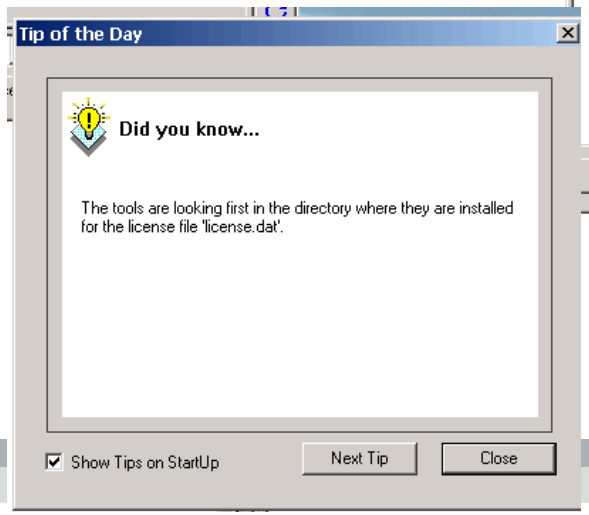
LAB1 -- Blinking LED

►Click “OK”

►Click “OK”

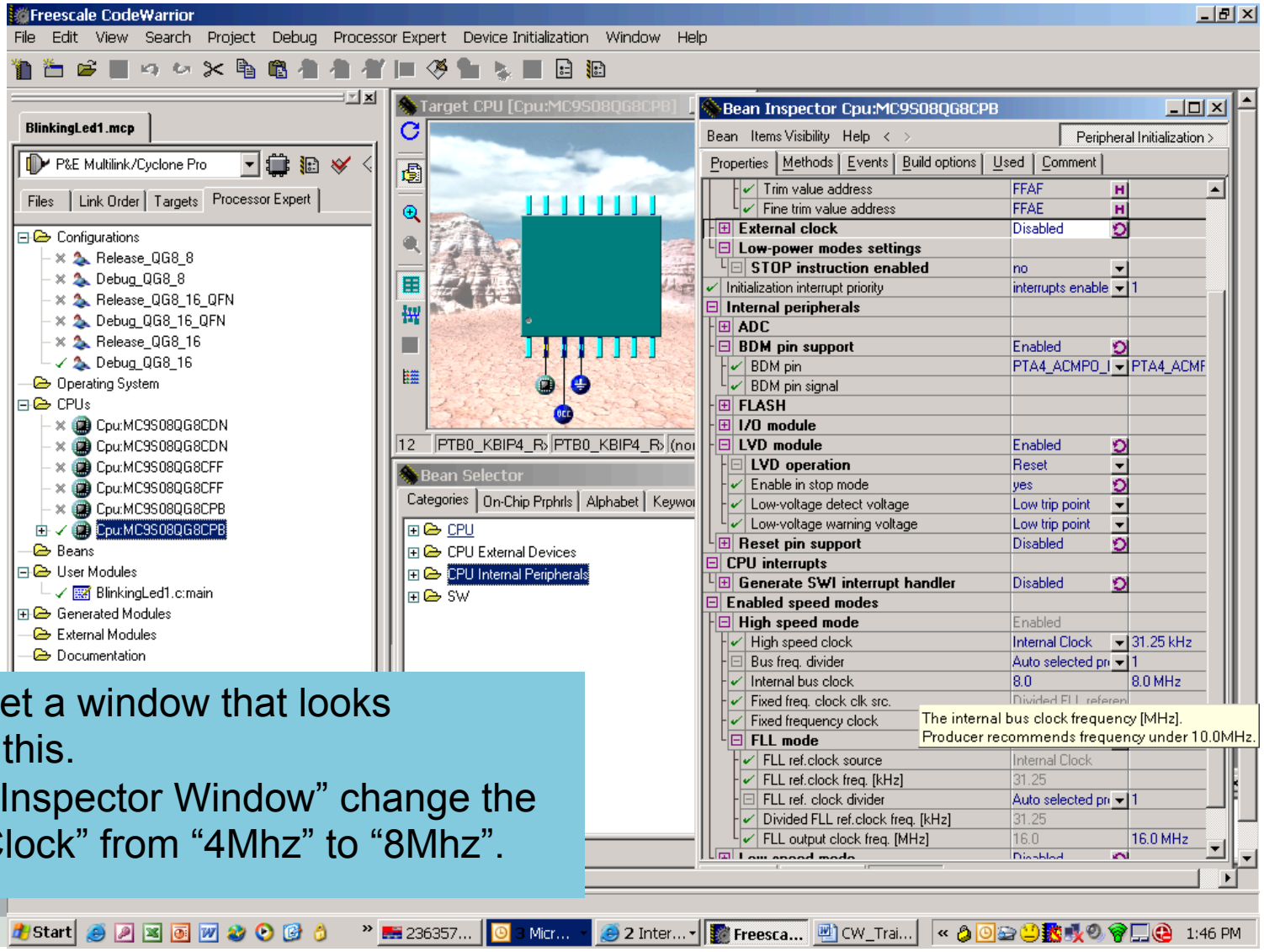


►Click “Close”



freescale semiconductor

LAB1 -- Blinking LED

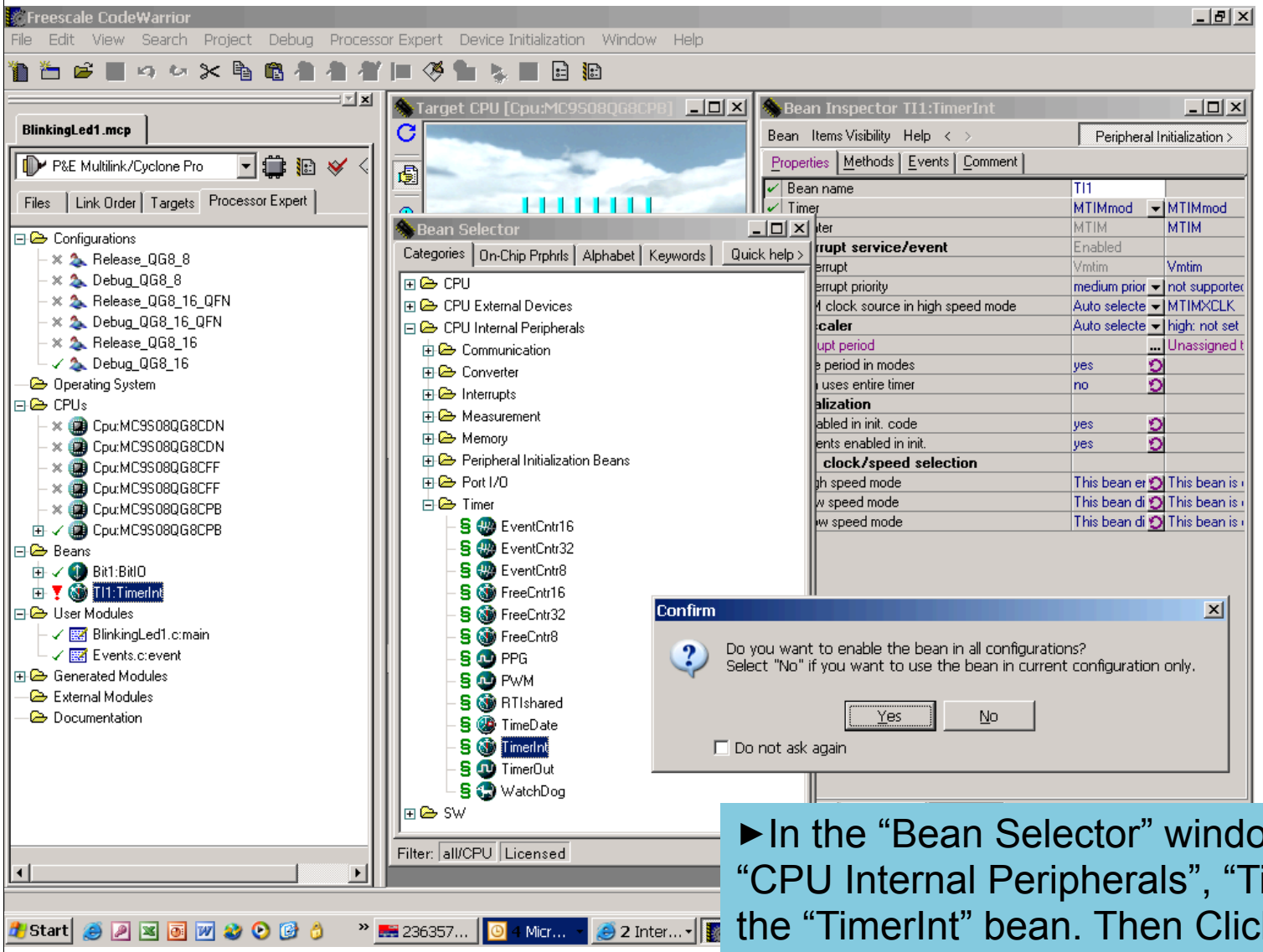


►You should get a window that looks somewhat like this.

►In the “Bean Inspector Window” change the “Internal Bus Clock” from “4Mhz” to “8Mhz”.

freescale semiconductor

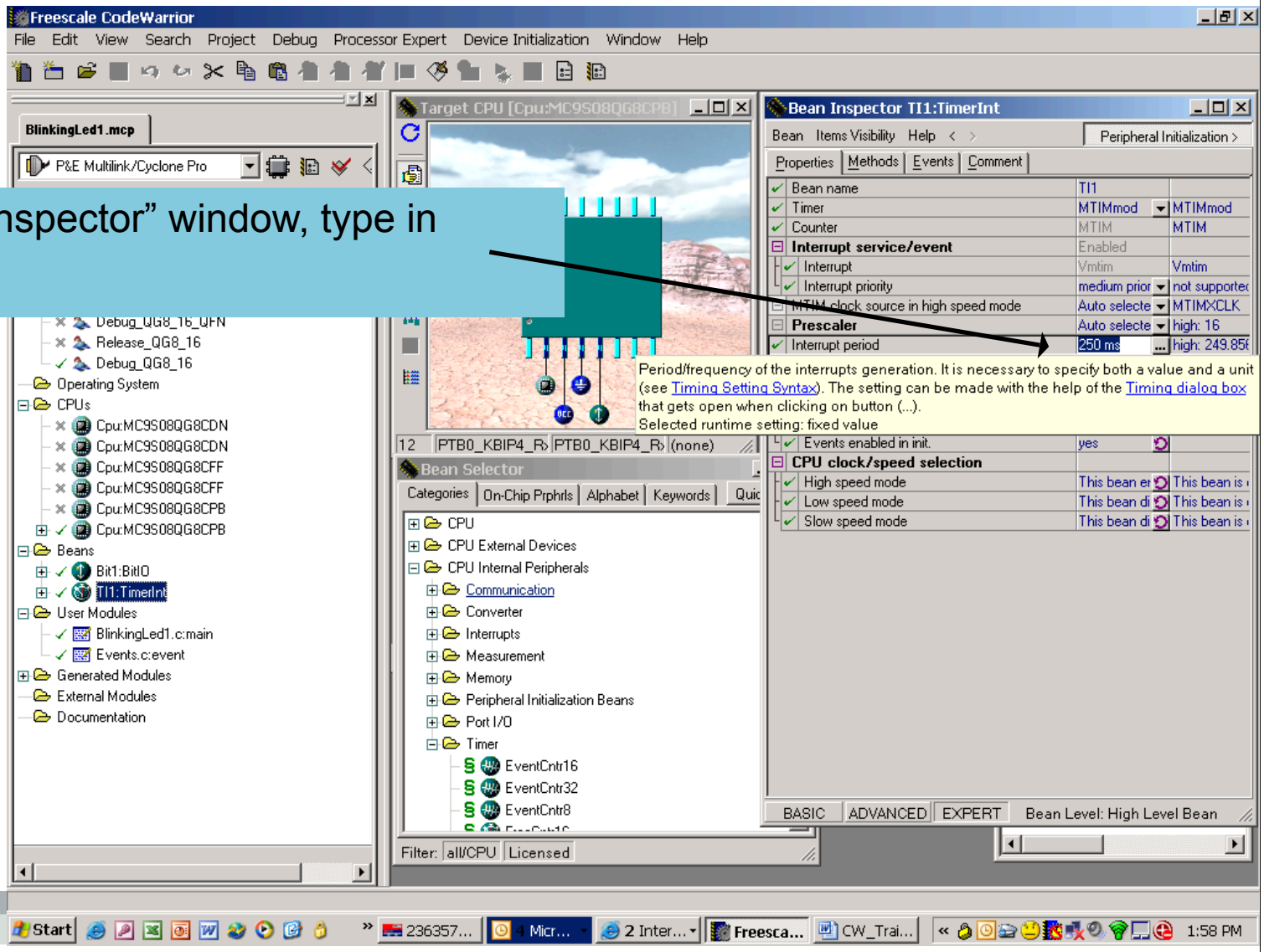
LAB1 -- Blinking LED



►In the “Bean Selector” window, expand the “CPU Internal Peripherals”, “Timer”, and select the “TimerInt” bean. Then Click “Yes” to confirm.

LAB1 -- Blinking LED

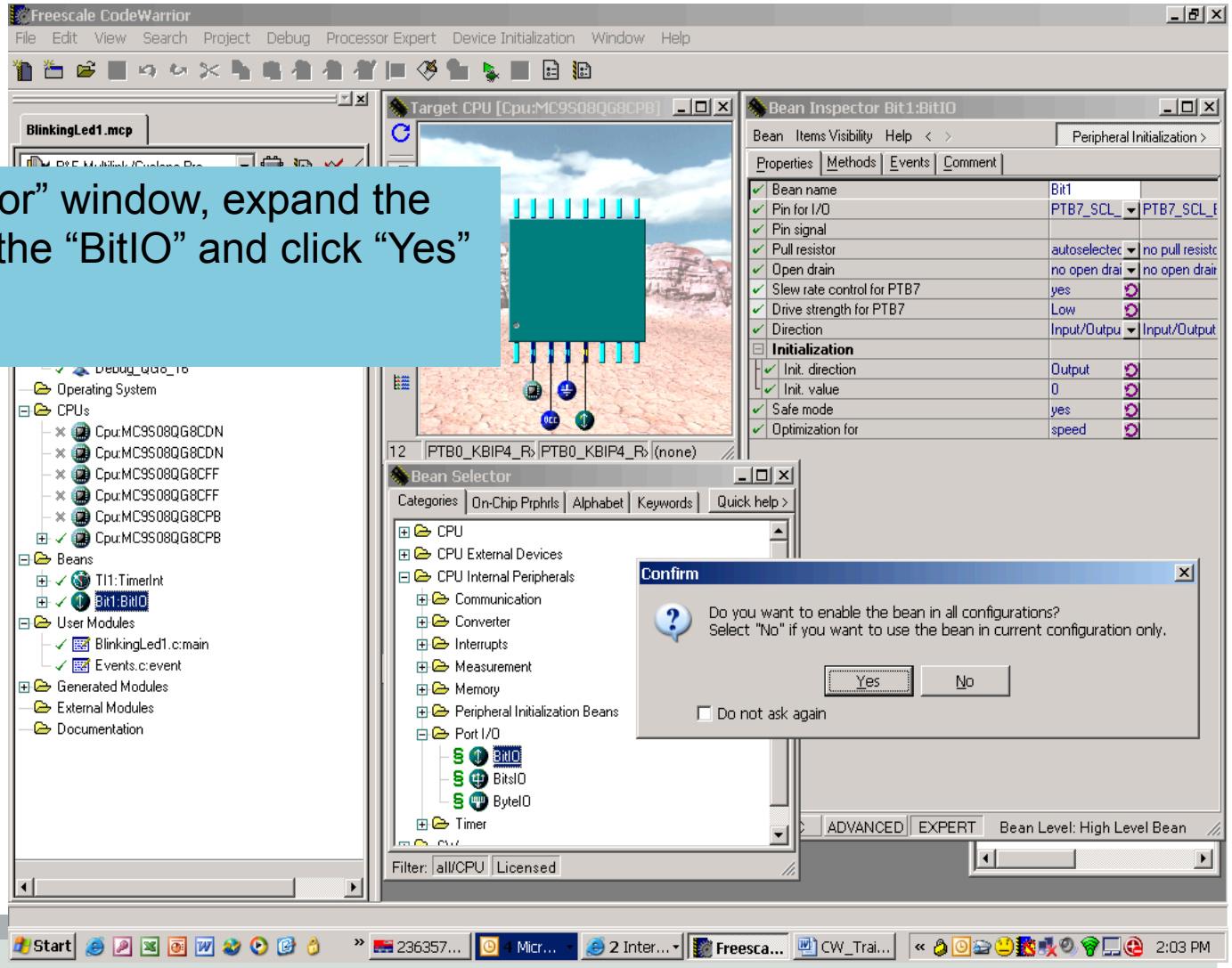
►In the “Bean Inspector” window, type in “250”(mSec)



freescale semiconductor

LAB1 -- Blinking LED

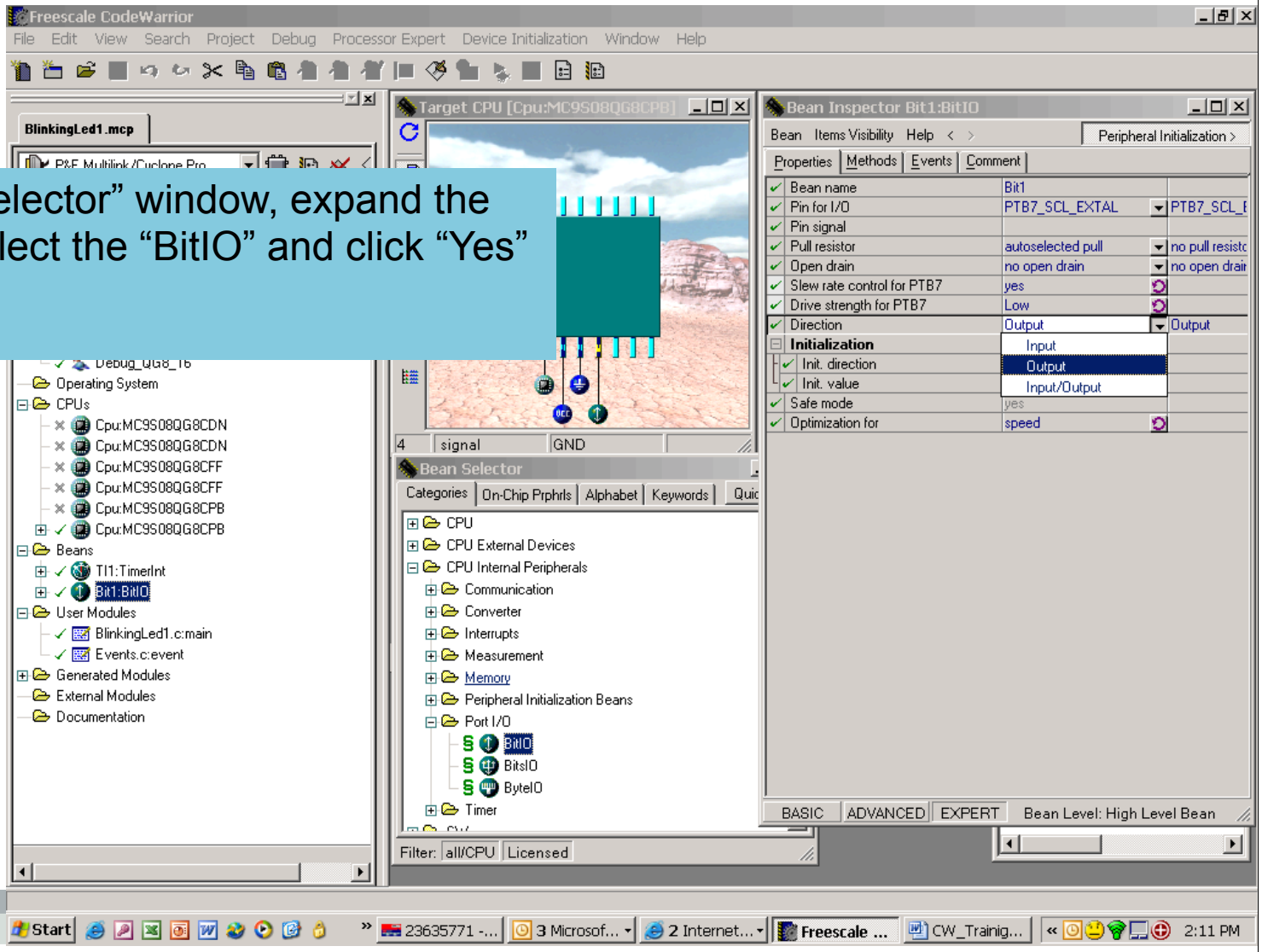
►In the “Bean Selector” window, expand the “Port I/O” and select the “BitIO” and click “Yes” to confirm.



freescale semiconductor

LAB1 -- Blinking LED

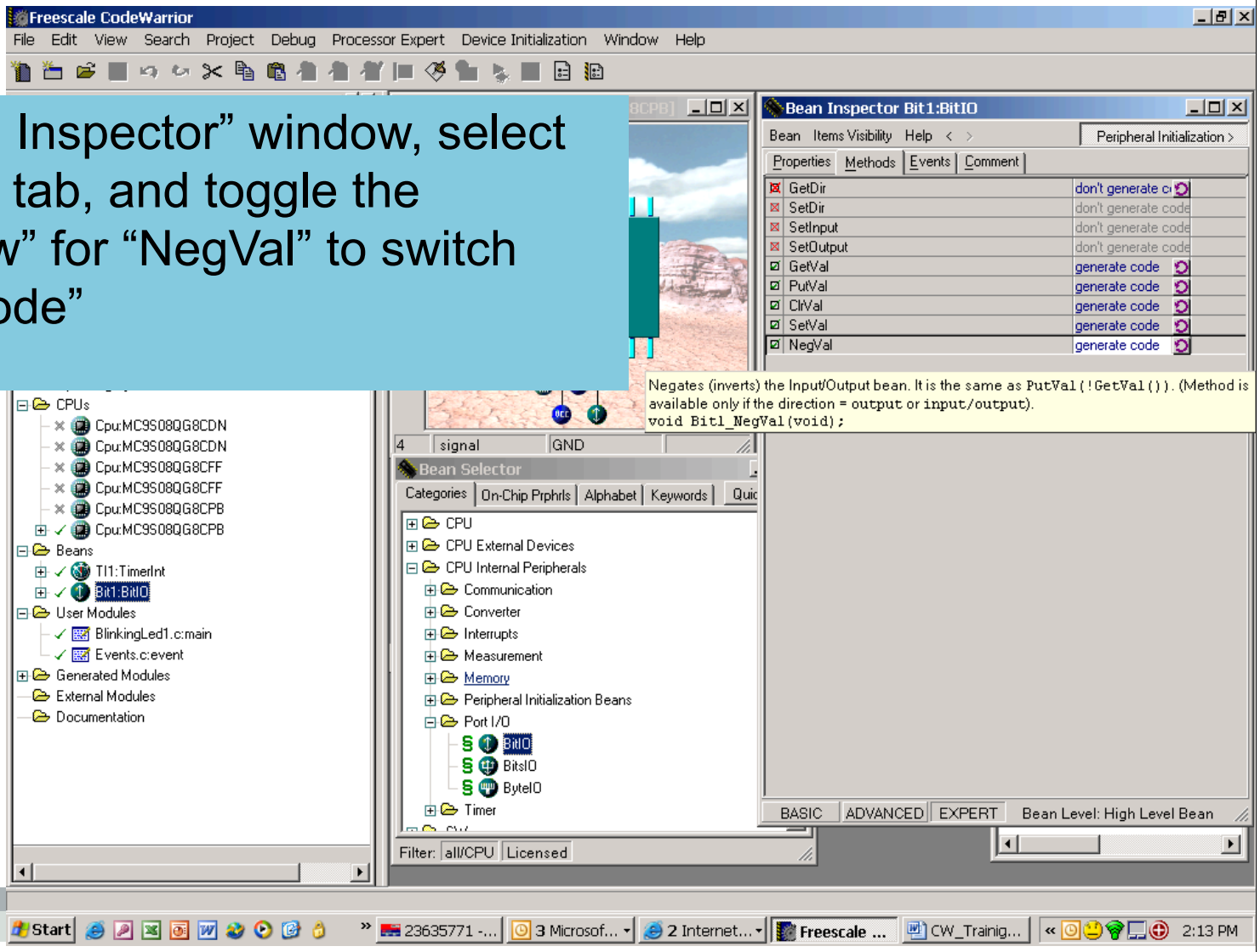
►In the “Bean Selector” window, expand the “Port I/O” and select the “BitIO” and click “Yes” to confirm.



freescale semiconductor

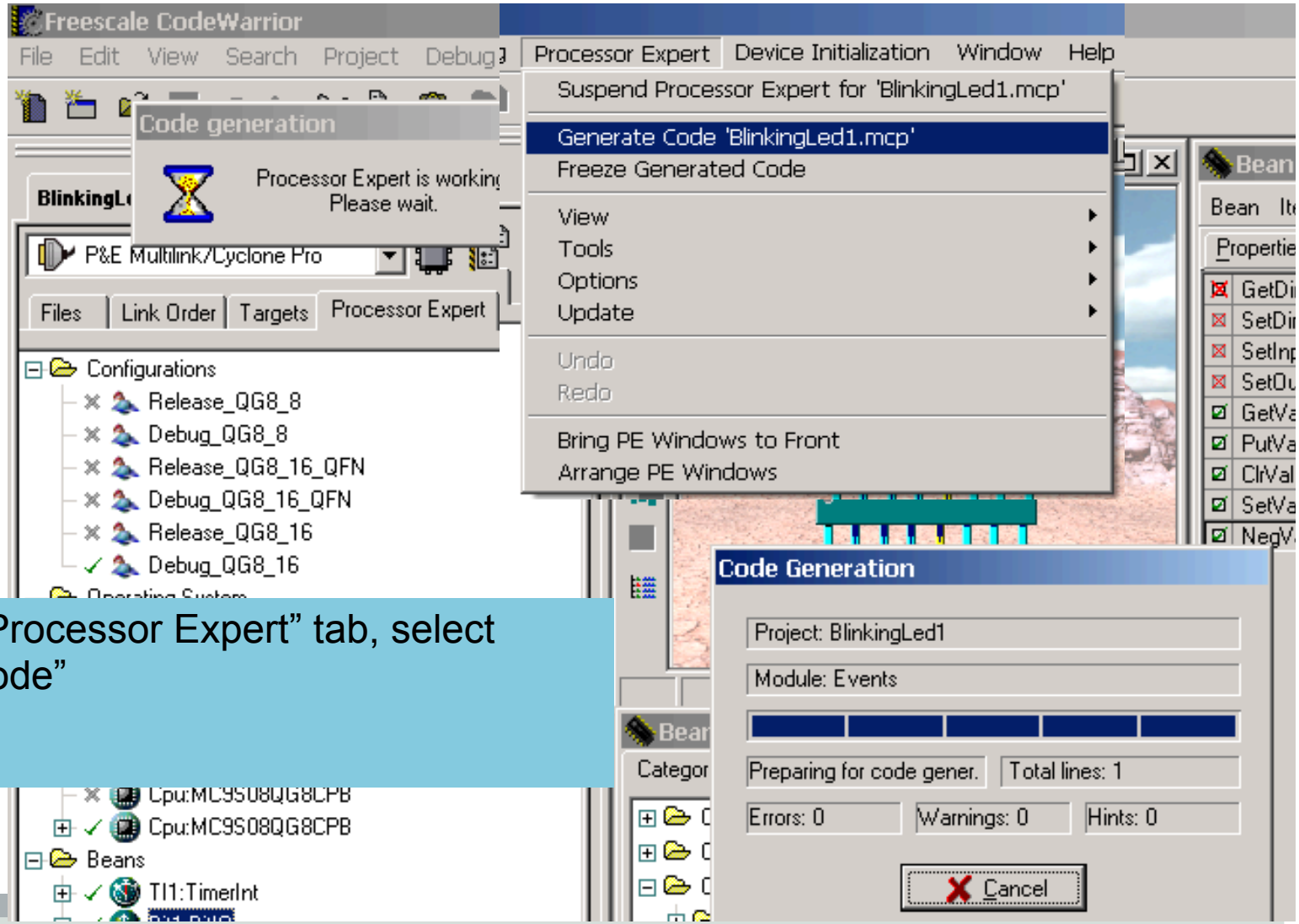
LAB1 -- Blinking LED

►In the “Bean Inspector” window, select the “Methods” tab, and toggle the “Circular Arrow” for “NegVal” to switch to“generate code”



freescale semiconductor

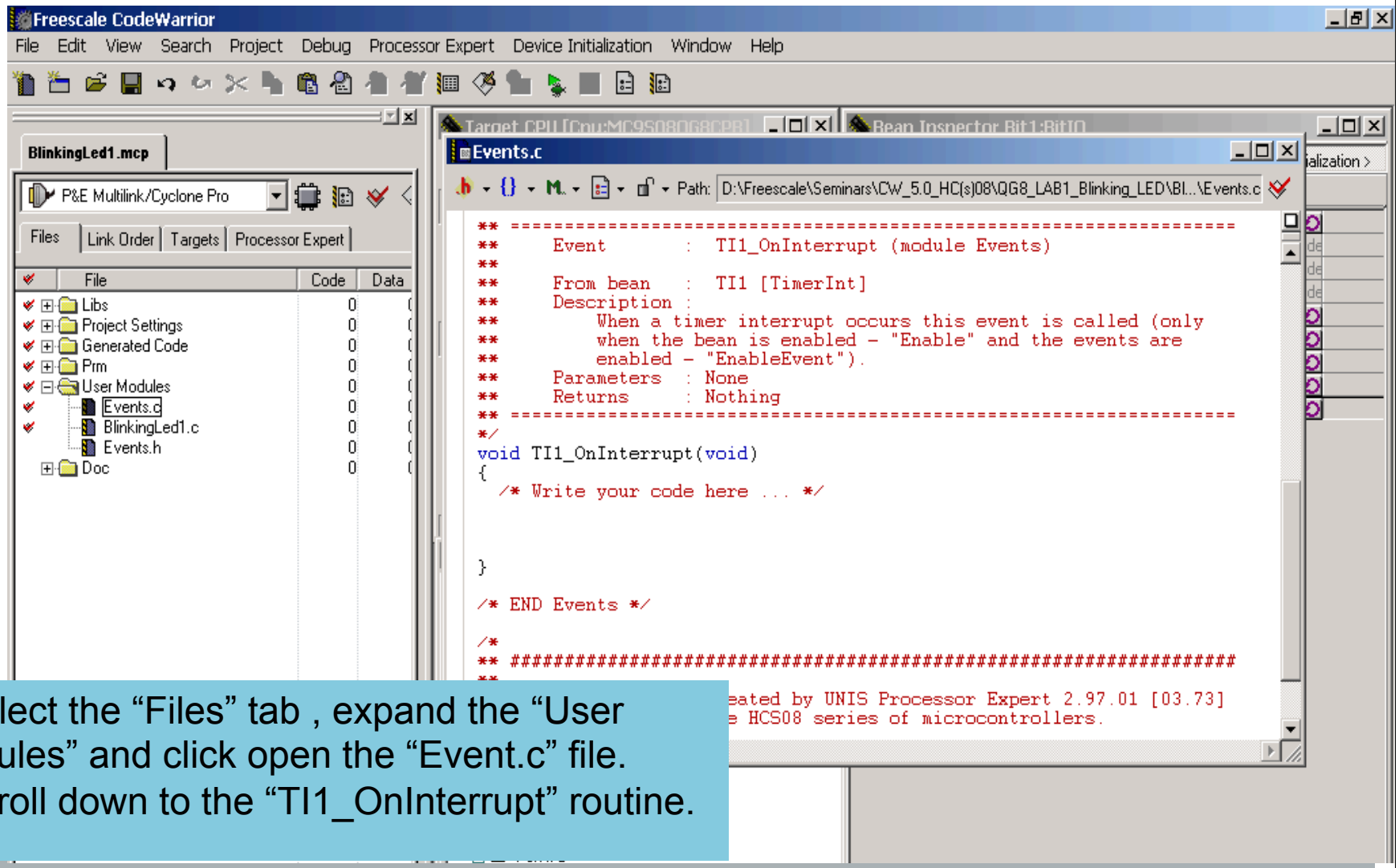
LAB1 -- Blinking LED



►From the “Processor Expert” tab, select “Generate Code”

freescale semiconductor

LAB1 -- Blinking LED



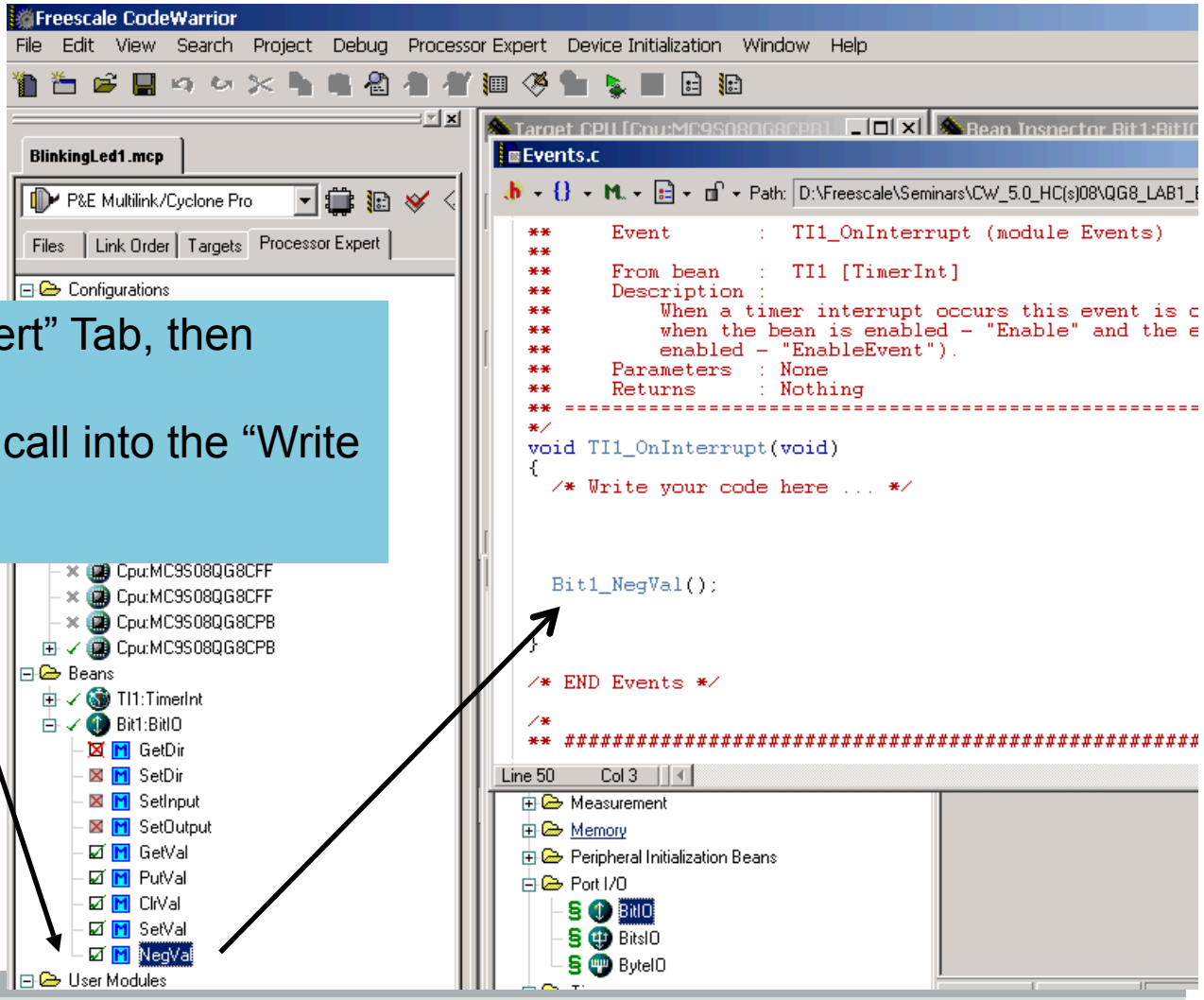
►Select the “Files” tab , expand the “User Modules” and click open the “Event.c” file.
►Scroll down to the “TI1_OnInterrupt” routine.

freescale semiconductor

LAB1 -- Blinking LED

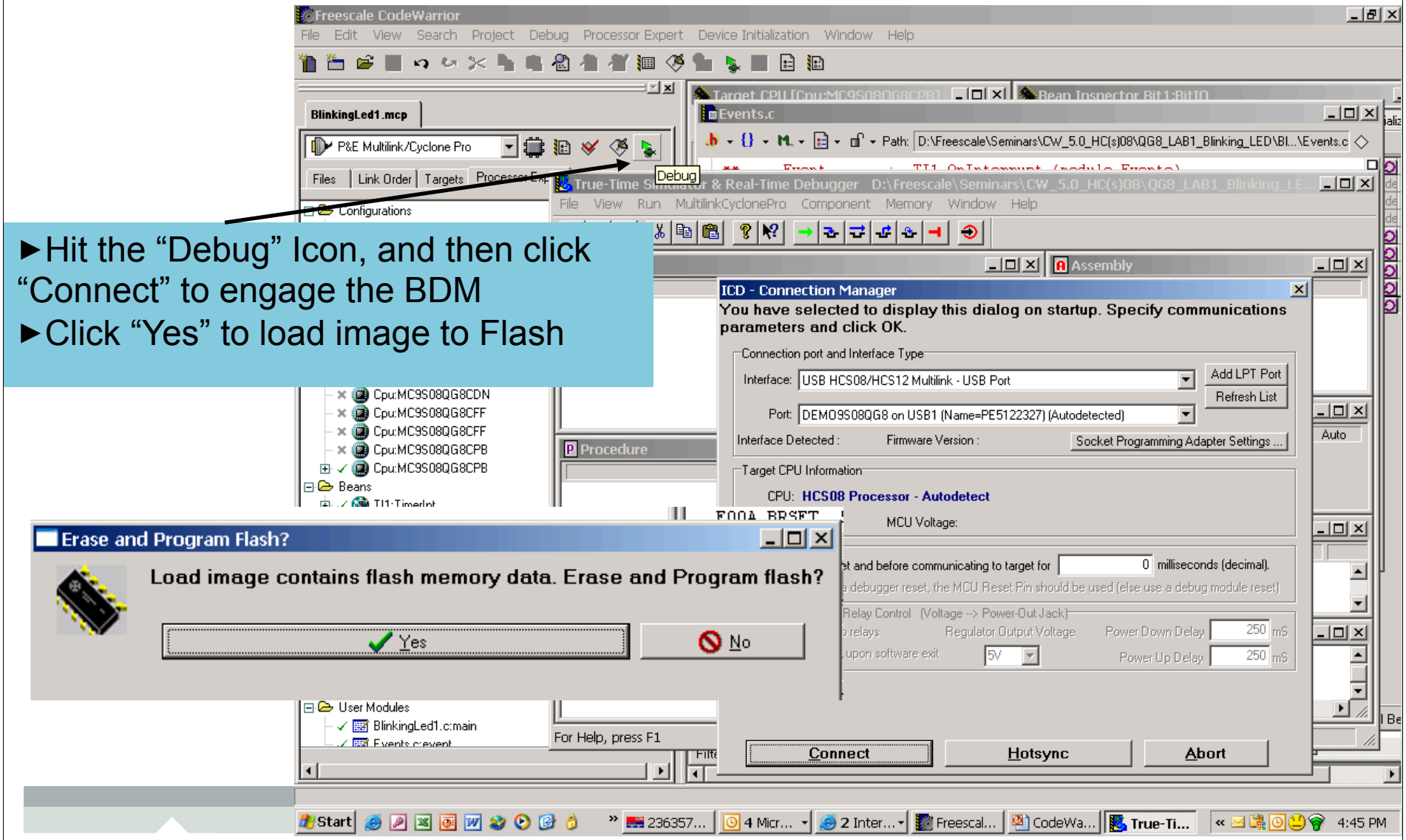
►Select the “Processor Expert” Tab, then expand the “BitIO” bean.

►Drop n Drag the “NegVal” call into the “Write your code here” section.



freescale semiconductor

LAB1 -- Blinking LED

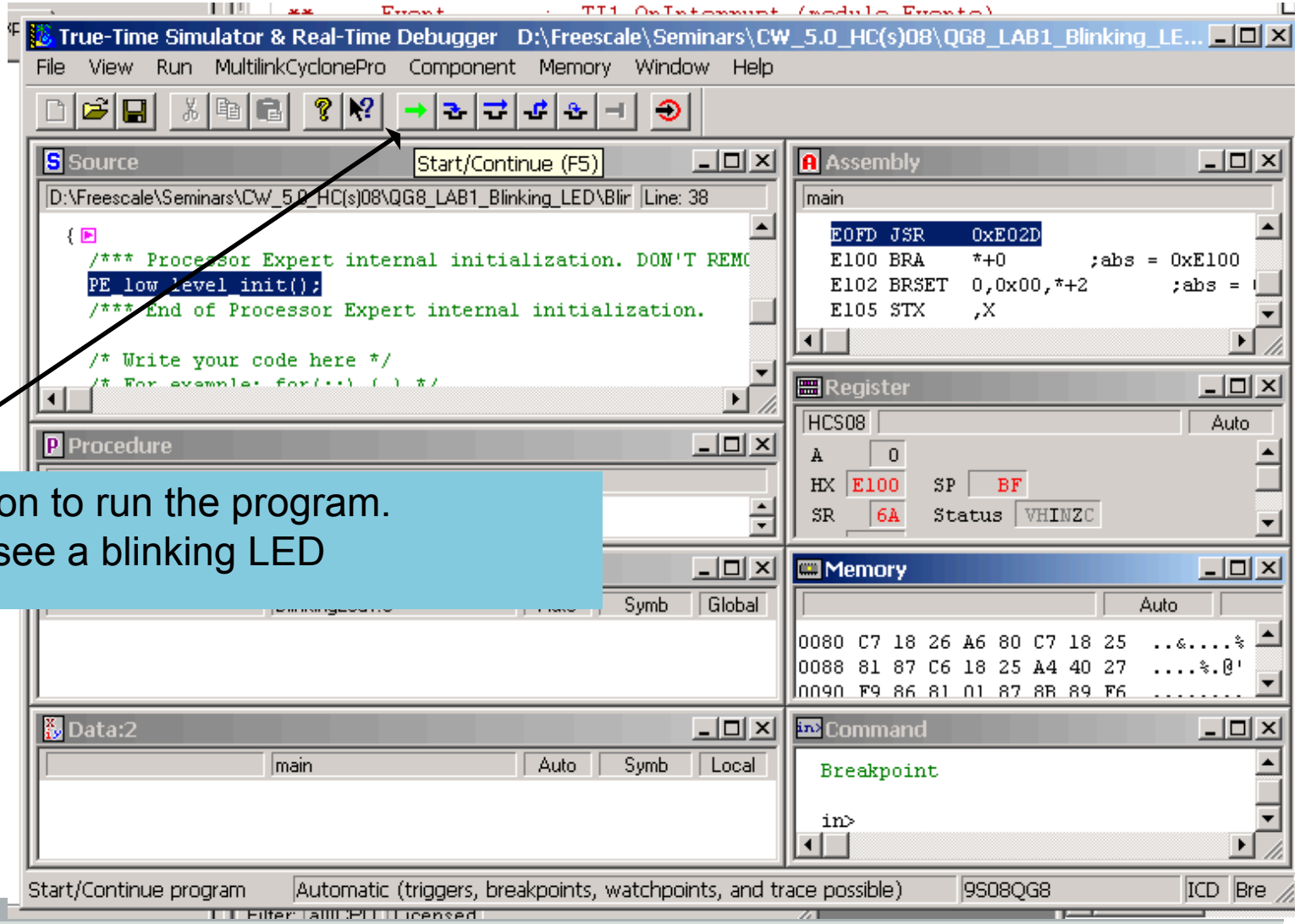


►Hit the “Debug” Icon, and then click “Connect” to engage the BDM

►Click “Yes” to load image to Flash

freescale semiconductor

LAB1 -- Blinking LED



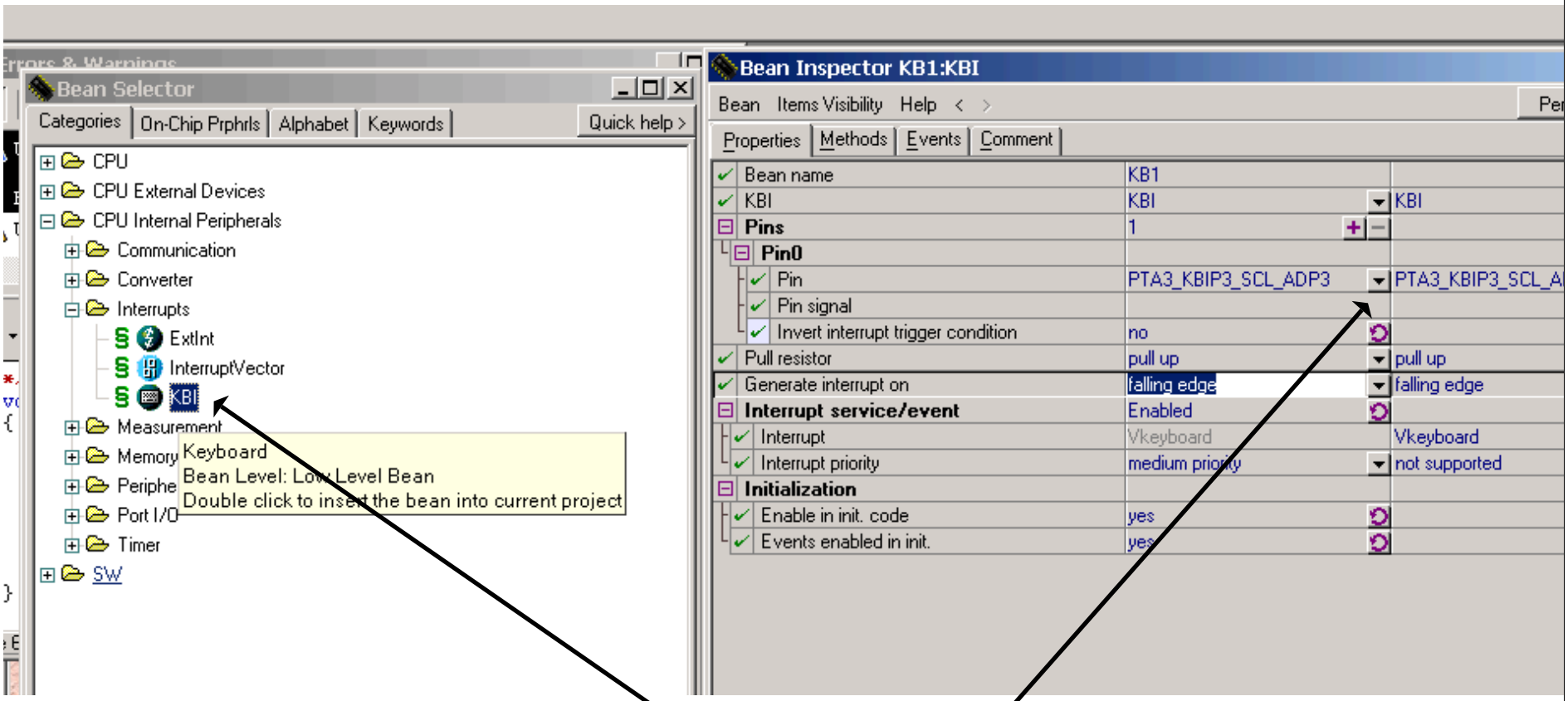
- ►Hit “Start” icon to run the program.
- ►You should see a blinking LED

freescale semiconductor

LAB 2 – KBI & ACMP

- Initialized KBI on SW2(PORTA3)
- Use SW2 to toggle LED1(PORTB6).
- Initialized Analog Comparator
- To use Photolight sensor to toggle LED2(PORTB6)

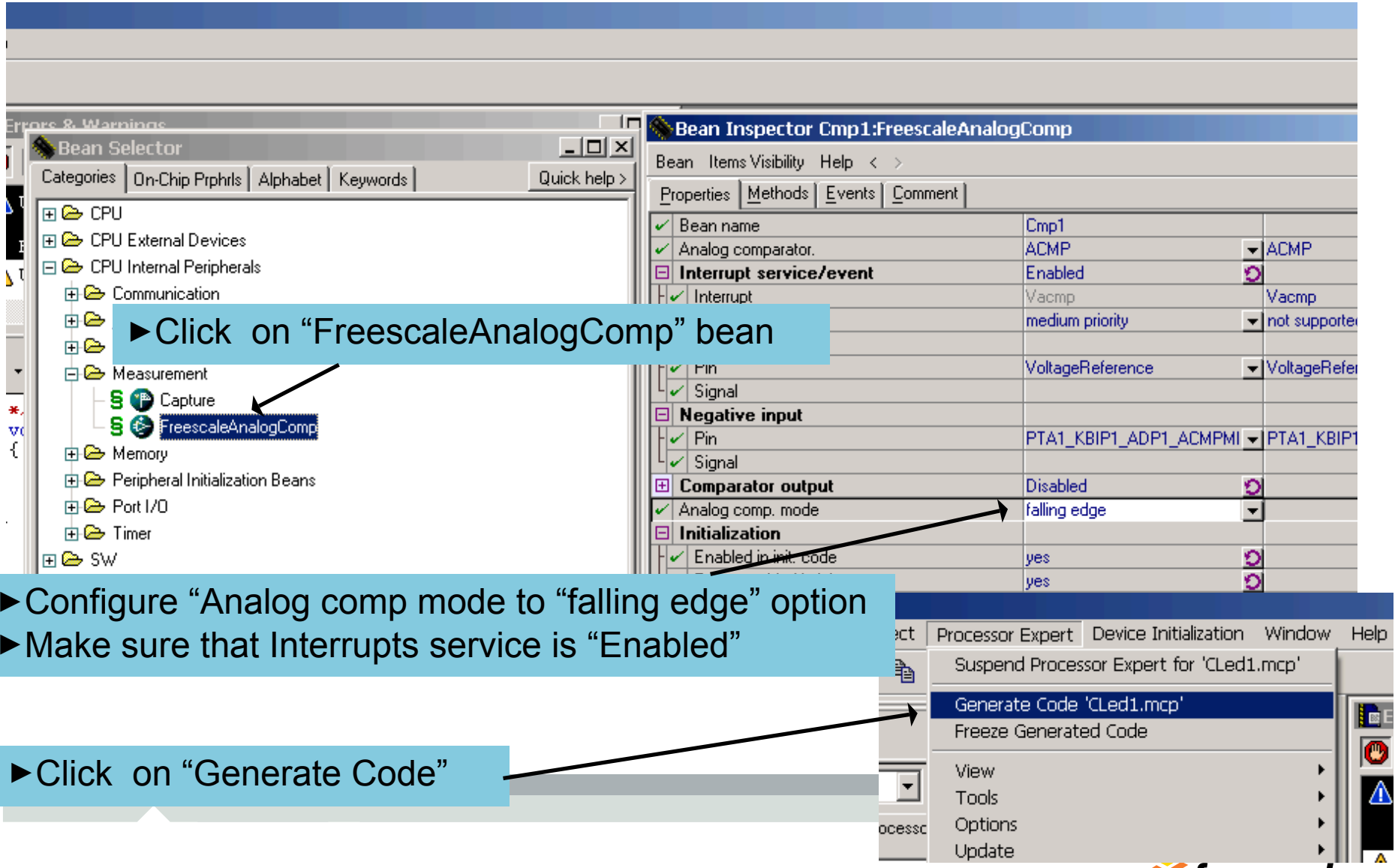
LAB2 – KBI & ACMP



- ►From the “Bean Selector” click on “KBI” bean
- ►Configure the Pin 0 for the KBI to be “PTA3_KBIP3_SCL_ADP3”
- ►Select “Pull Up” and “fallling edge” options .
- ►Make sure that Interrupts service is “Enabled”

freescale semiconductor

LAB2 – KBI & ACMP



►Click on “FreescaleAnalogComp” bean

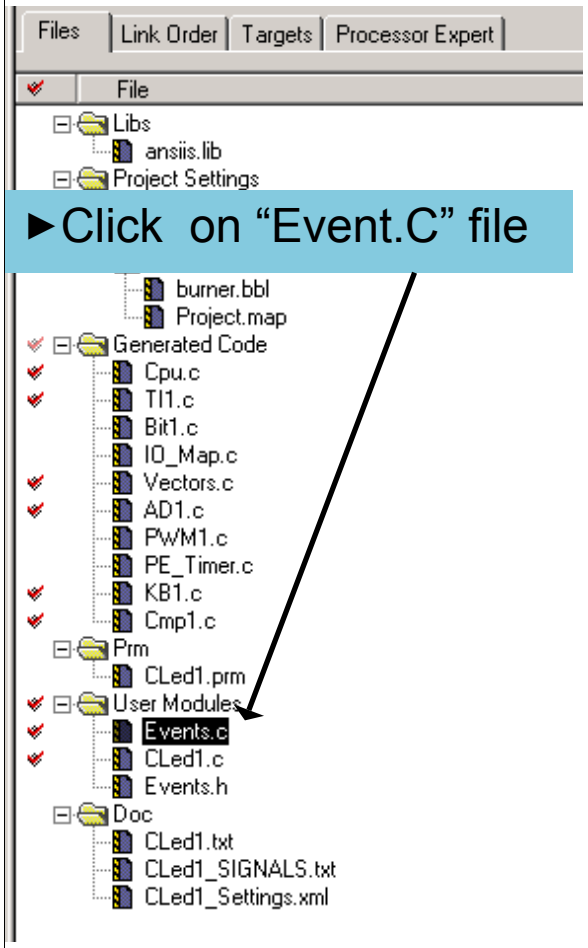
►Configure “Analog comp mode to “falling edge” option

►Make sure that Interrupts service is “Enabled”

►Click on “Generate Code”

freescale semiconductor

LAB2 – KBI & ACMP



►Click on “Event.C” file

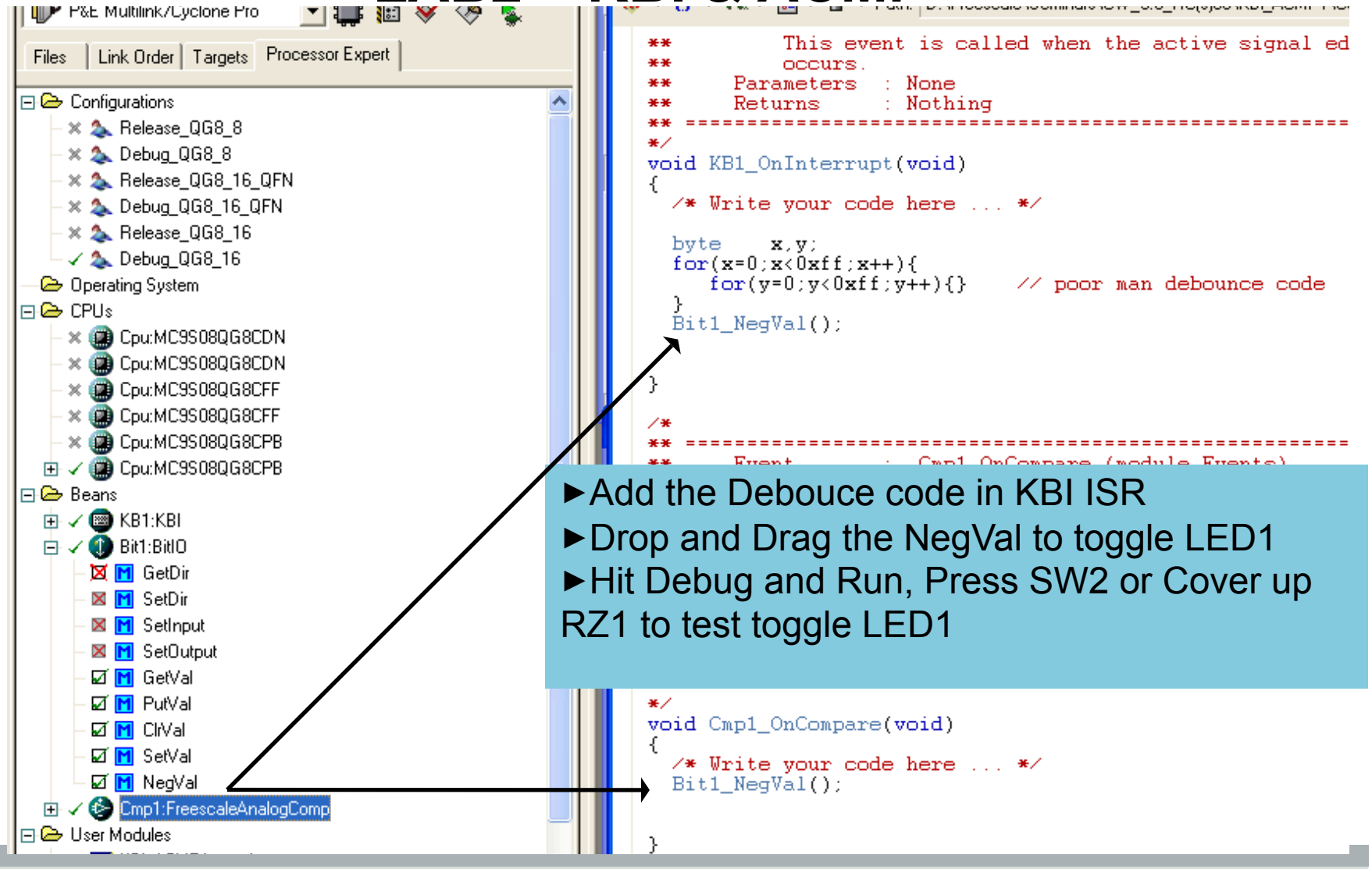
```
**         comparator output. Interr
**         enabled for this event to
**     Parameters  : None
**     Returns     : Nothing
** ===================================
*/
void Cmp1_OnCompare(void)
{
  /* Write your code here ... */
}

/*
** ===================================
**     Event       :  KB1_OnInterrup
**
**     From bean   :  KB1 [KBI]
**     Description :
**         This event is called when
**         occurs.
**     Parameters  : None
**     Returns     : Nothing
** ===================================
*/
void KB1_OnInterrupt(void)
{
  /* Write your code here ... */
}

/* END Events */
```

►Scroll down to these two Events

►This is where you would add the code to toggle LED1

freescale semiconductor

LAB2 – KBI & ACMP



- ►Add the Debouce code in KBI ISR
- ►Drop and Drag the NegVal to toggle LED1
- ►Hit Debug and Run, Press SW2 or Cover up RZ1 to test toggle LED1

freescale semiconductor

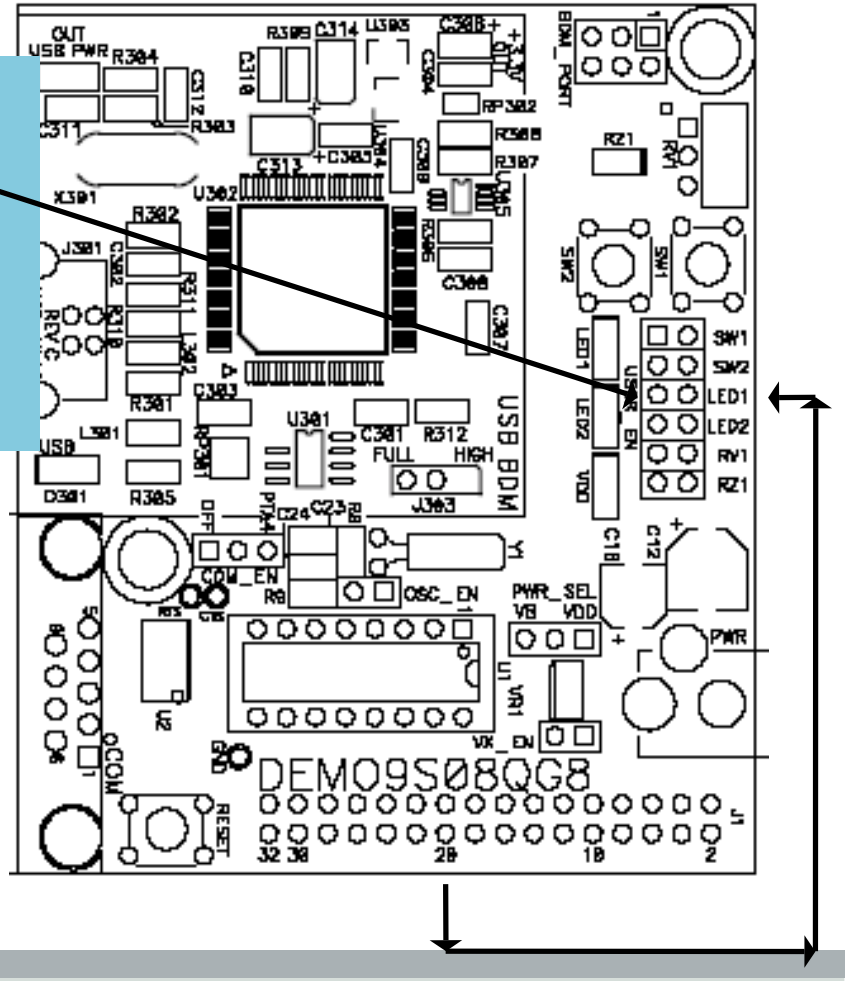
LAB3 – ADC & PWM

- Initialize ADC Channel 0 (PORTA0 –Potentiometer)
- Initialize PWM for 1mSec period with 64uSec initial on Pulse.
- Use ADC reading to reconfigure PWM ratio (duty cycle).
- Re-route PWM output to an LED1 to show dimming effect.

LAB3 – ADC & PWM

►Remove Jumper on LED1

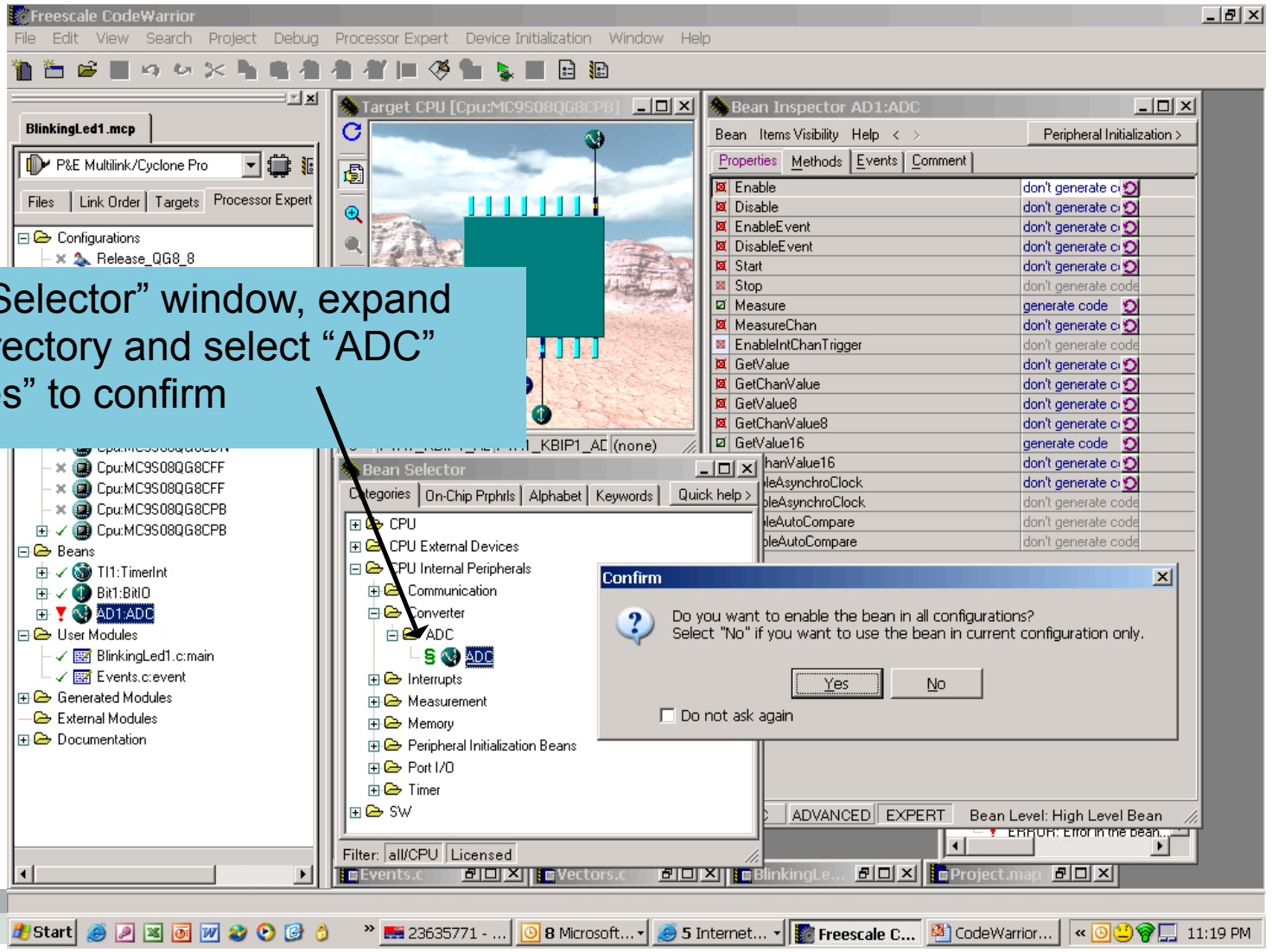
►Jumper a wire between
J1(23) and LED1(2)
LED1(2) is the pin closer to the edge of the board



freescale semiconductor

LAB3 – ADC & PWM

►From the “Bean Selector” window, expand the “Converter” directory and select “ADC” bean, and click “Yes” to confirm

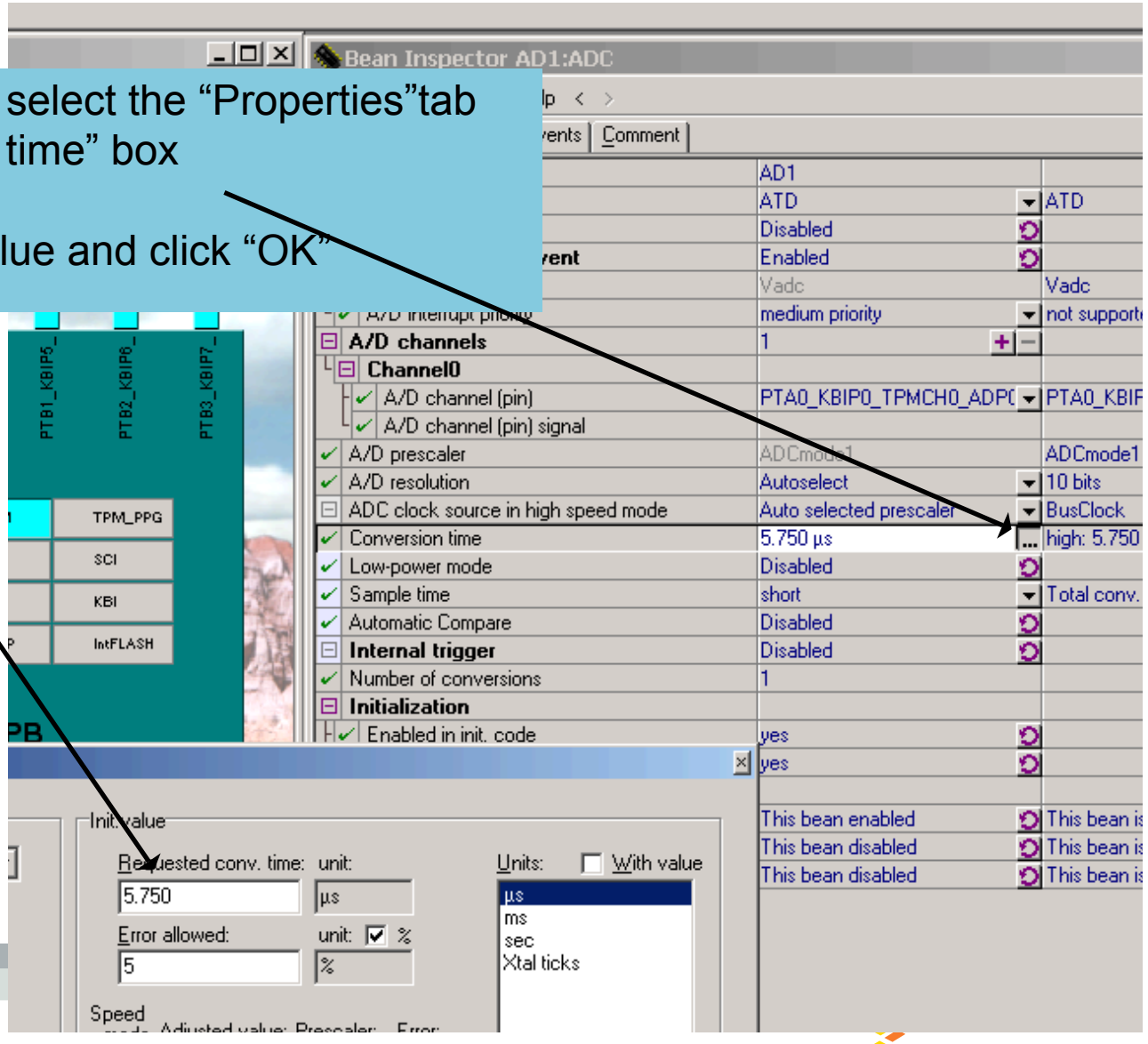


freescale semiconductor

LAB3 – ADC & PWM

►In the “Bean Inspector”, select the “Properties”tab and click the “Conversion time” box

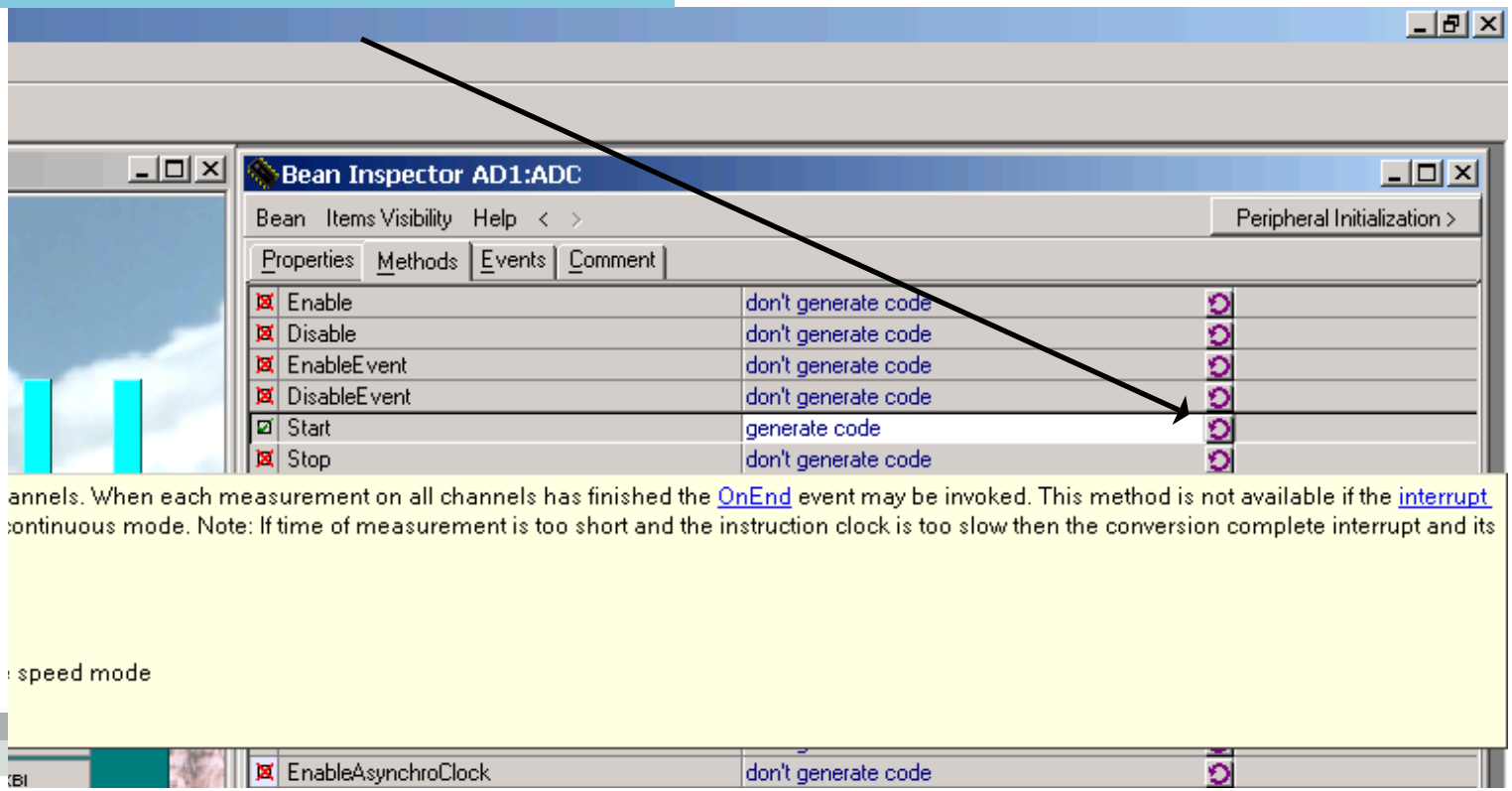
►Enter a “5.750” uSec value and click “OK”



freescale semiconductor

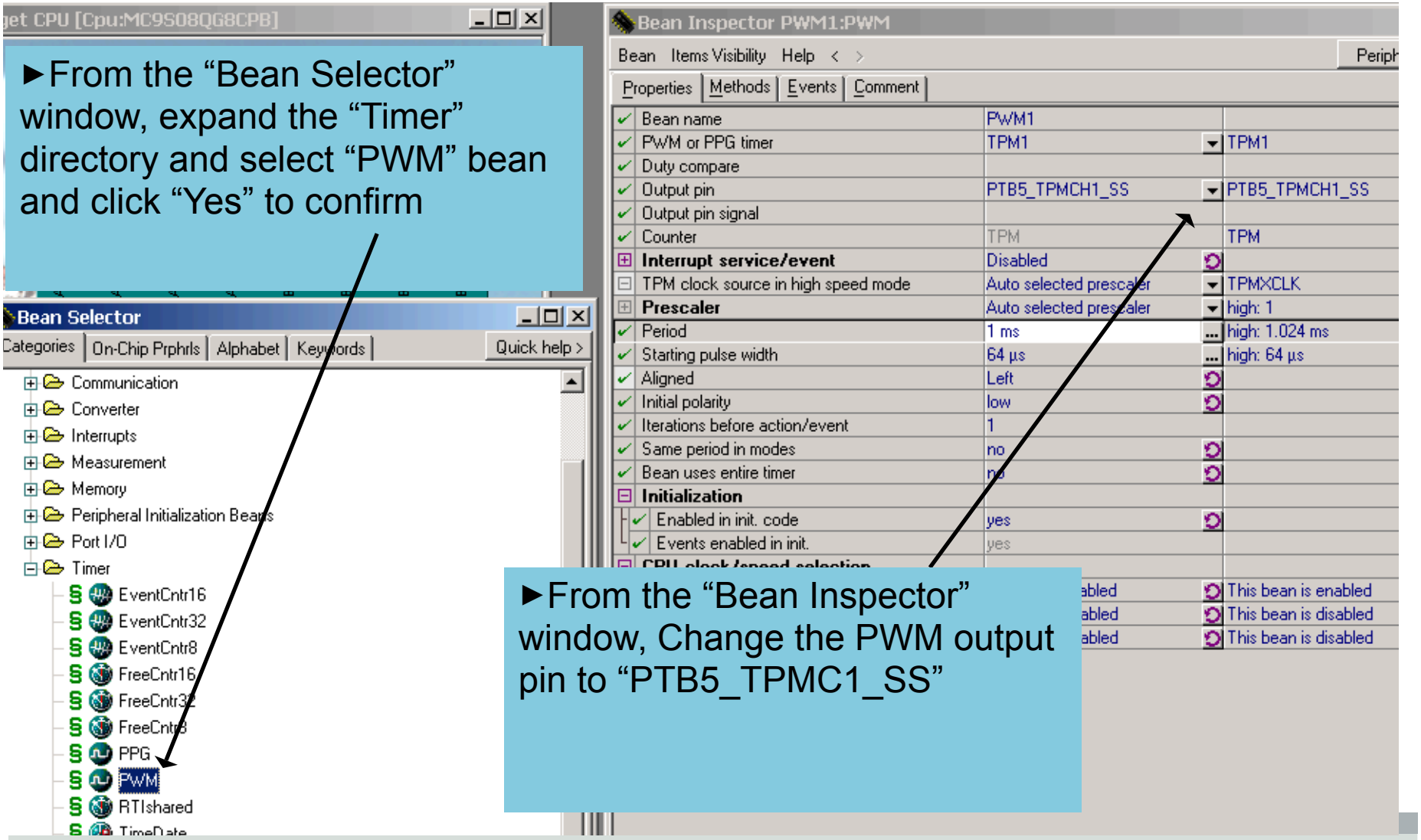
LAB3 – ADC & PWM

►In the “Bean Inspector”, select the “Properties”tab and toggle the “Start” method to indicate “generate code”



freescale semiconductor

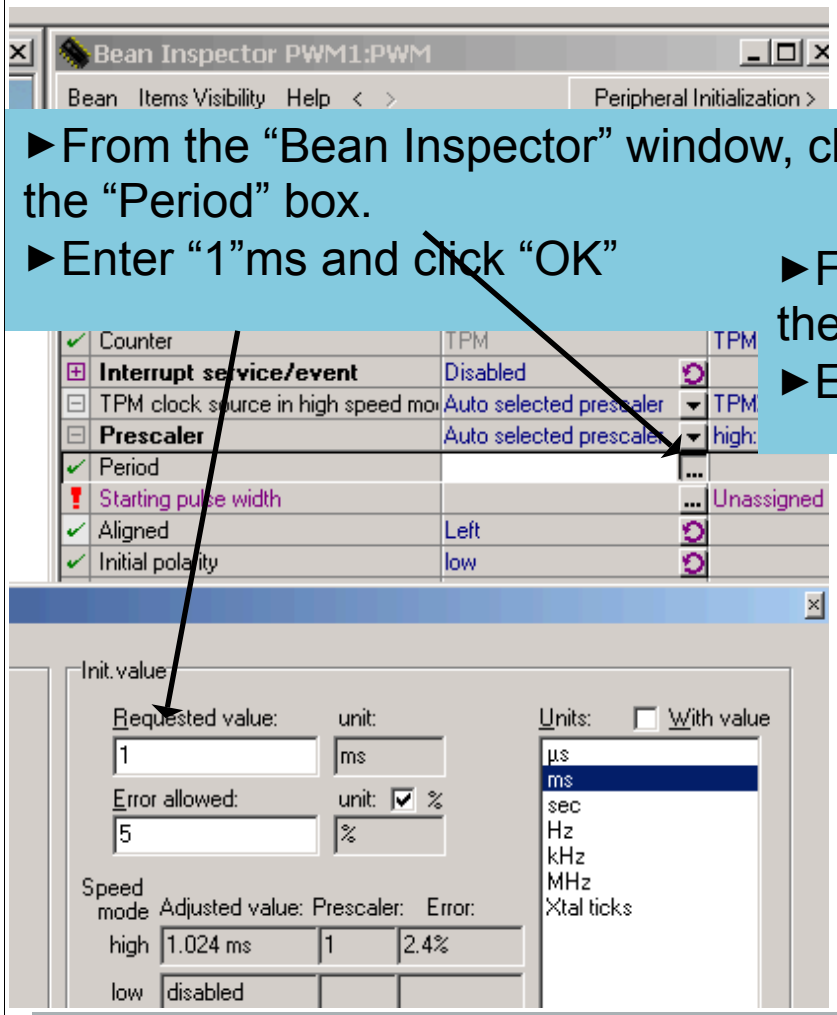
LAB3 – ADC & PWM



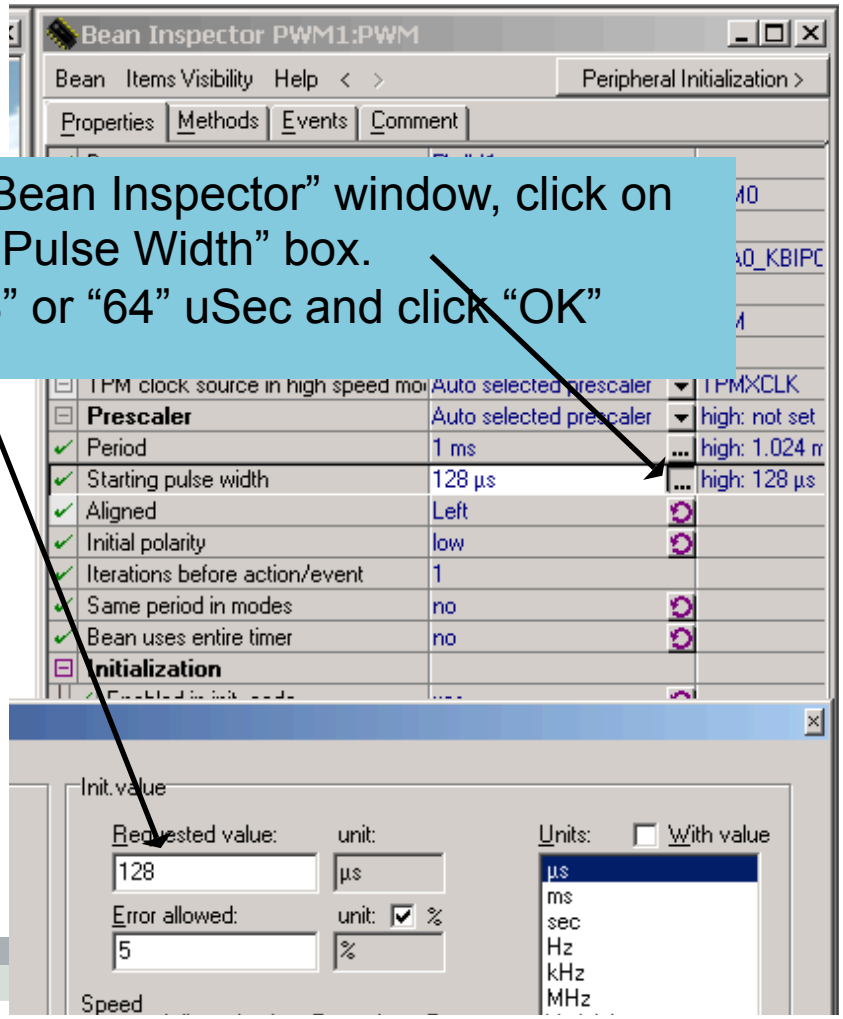
freescale semiconductor

LAB3 – ADC & PWM

►From the “Bean Inspector” window, click on the “Period” box.
►Enter “1”ms and click “OK”

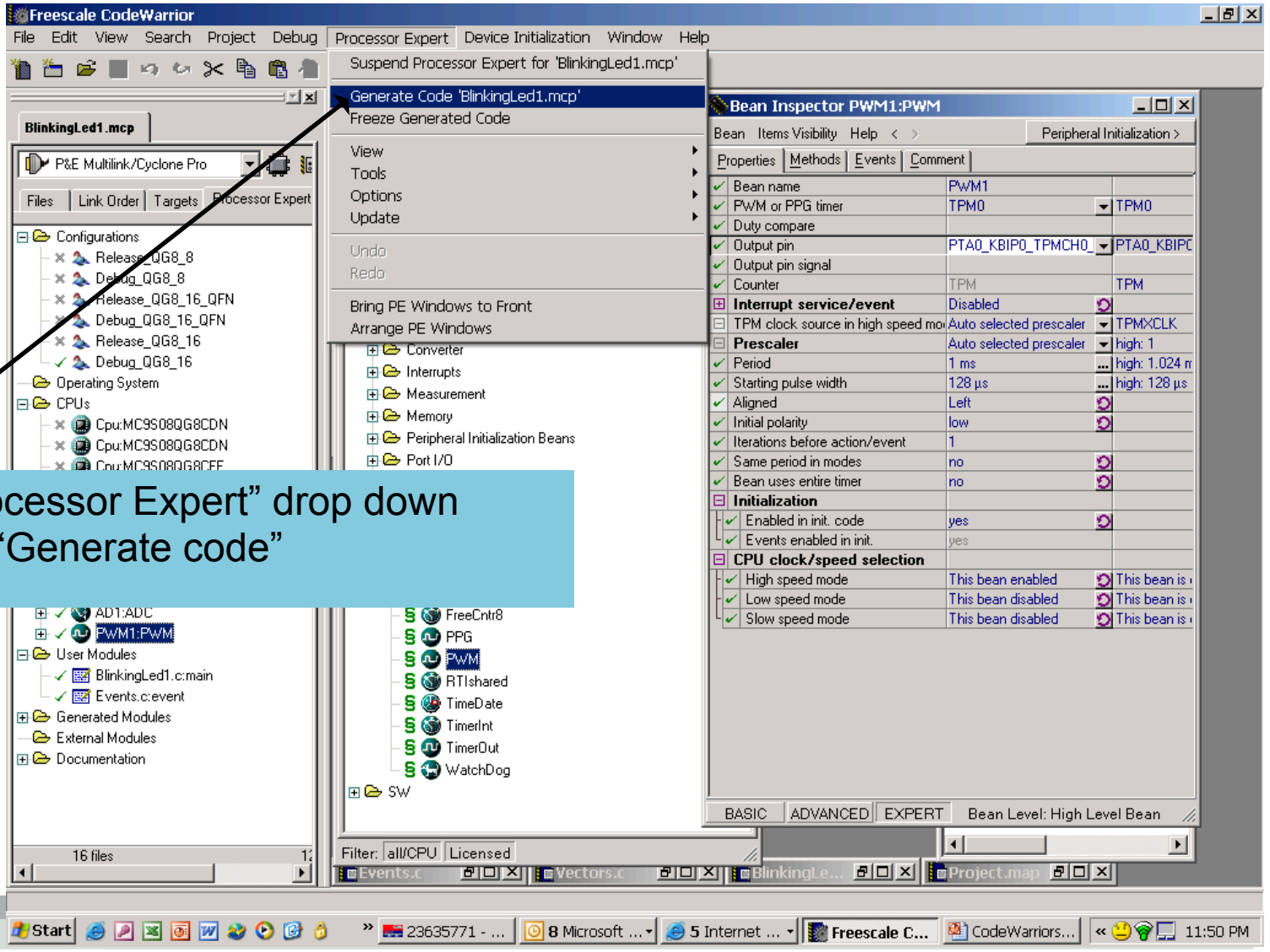


►From the “Bean Inspector” window, click on the “Starting Pulse Width” box.
►Enter “128” or “64” uSec and click “OK”



freescale semiconductor

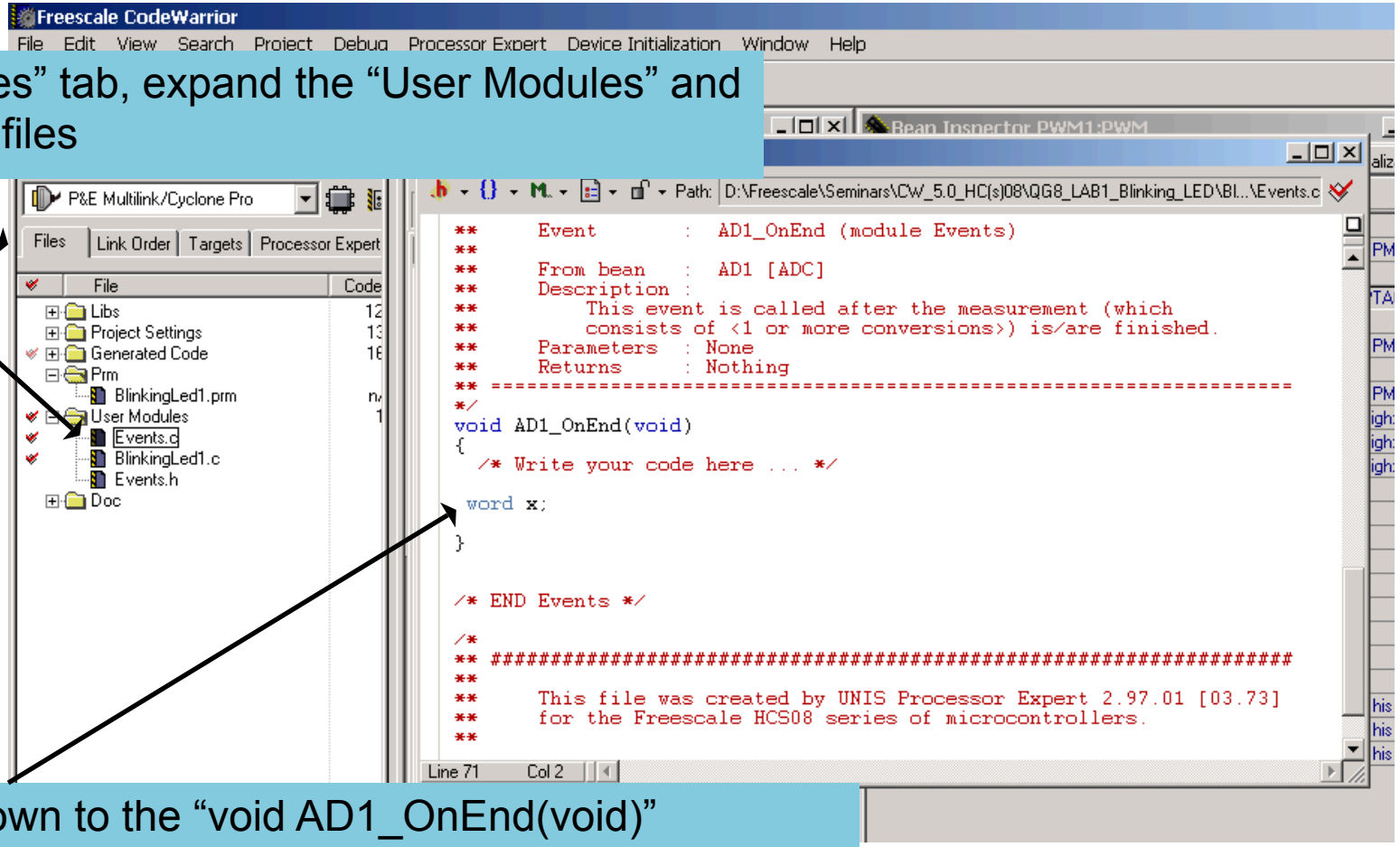
LAB3 – ADC & PWM



►From the “Processor Expert” drop down menu, click on “Generate code”

freescale semiconductor

LAB3 – ADC & PWM



►From the “Files” tab, expand the “User Modules” and open “Event.c” files

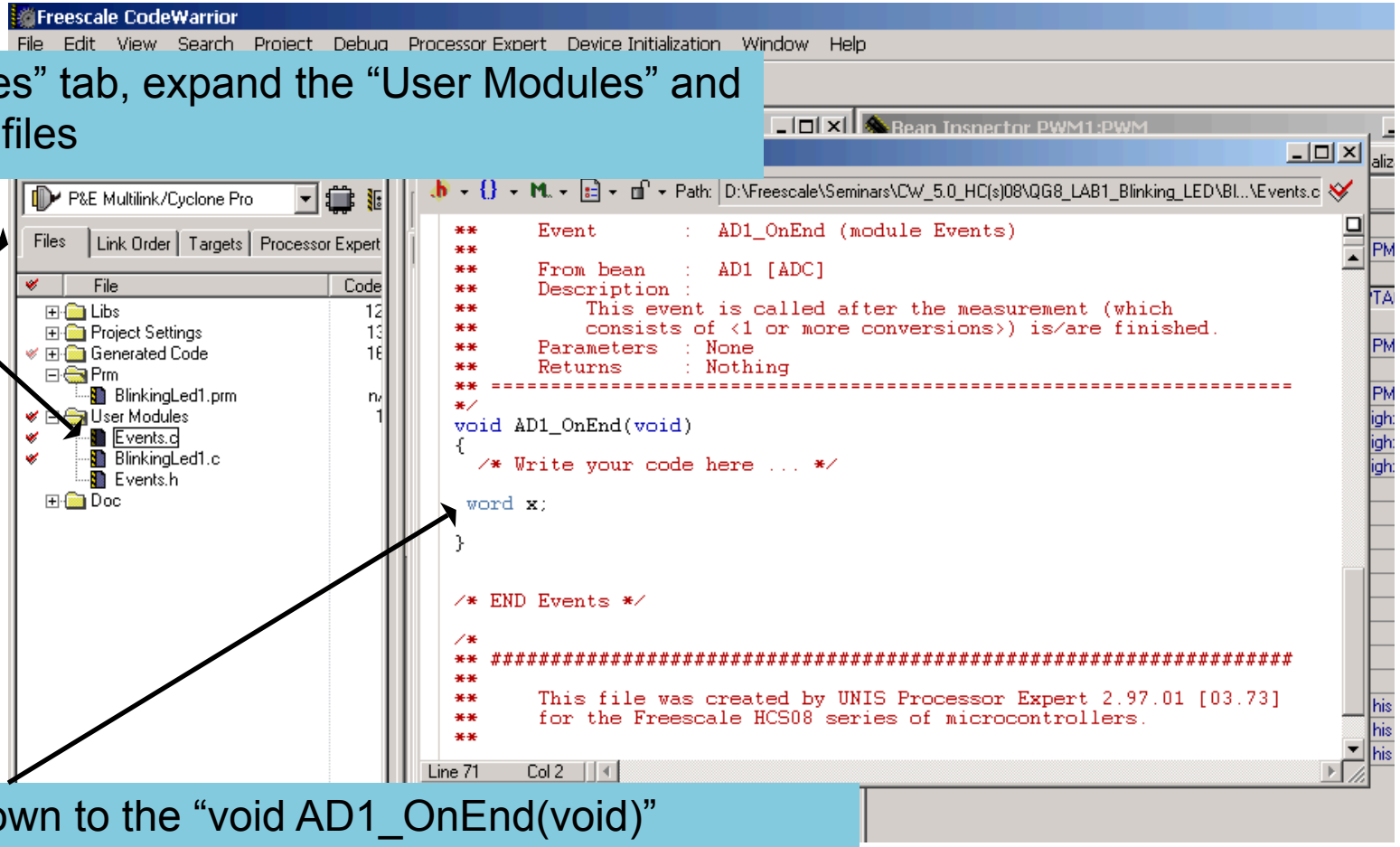
►Scroll down to the “void AD1_OnEnd(void)”

►Type “word x;” in the /*Write your code here*/

freescale semiconductor

LAB3 – ADC & PWM

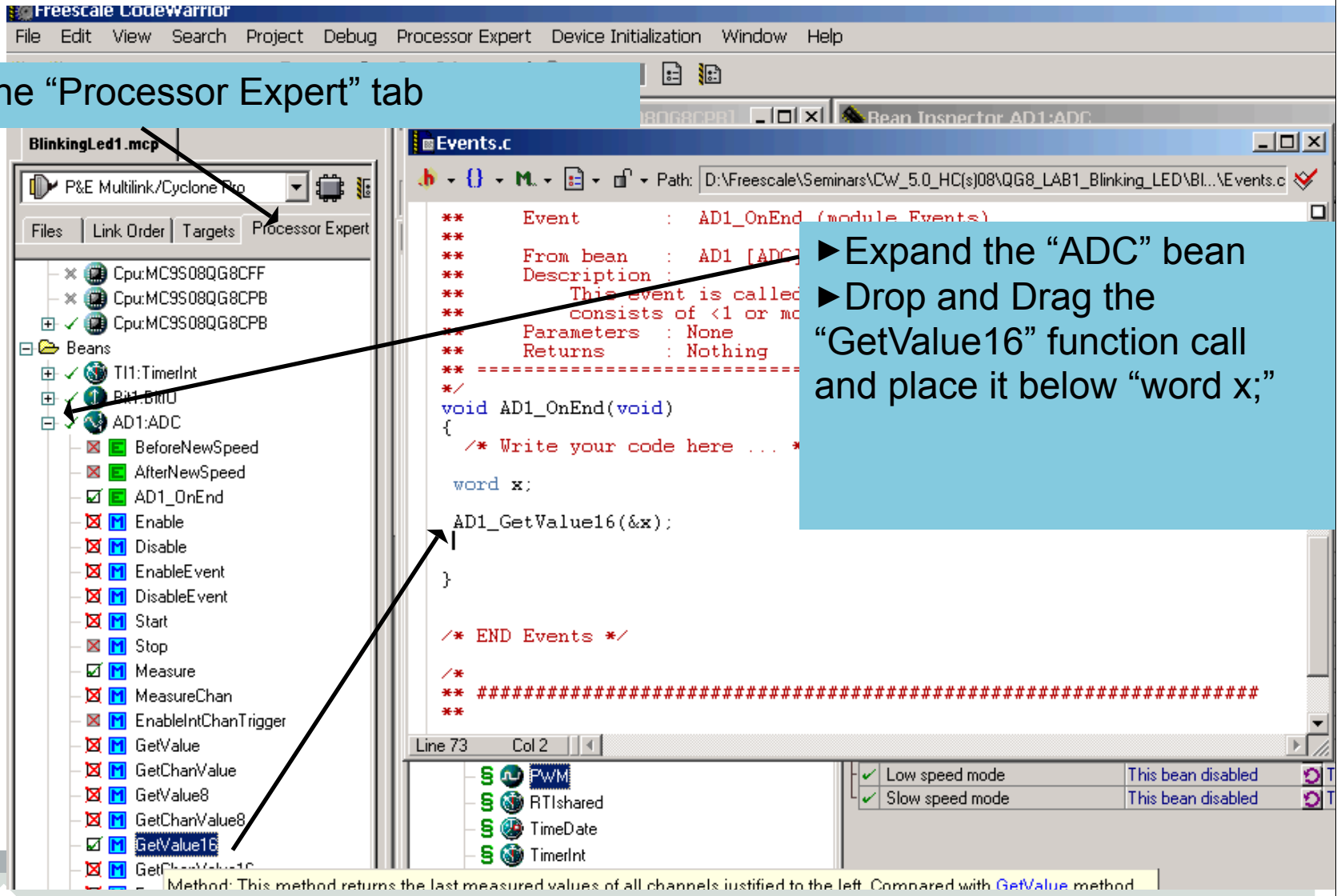
►From the “Files” tab, expand the “User Modules” and open “Event.c” files



►Scroll down to the “void AD1_OnEnd(void)”
►Type “word x;” in the /*Write your code here*/

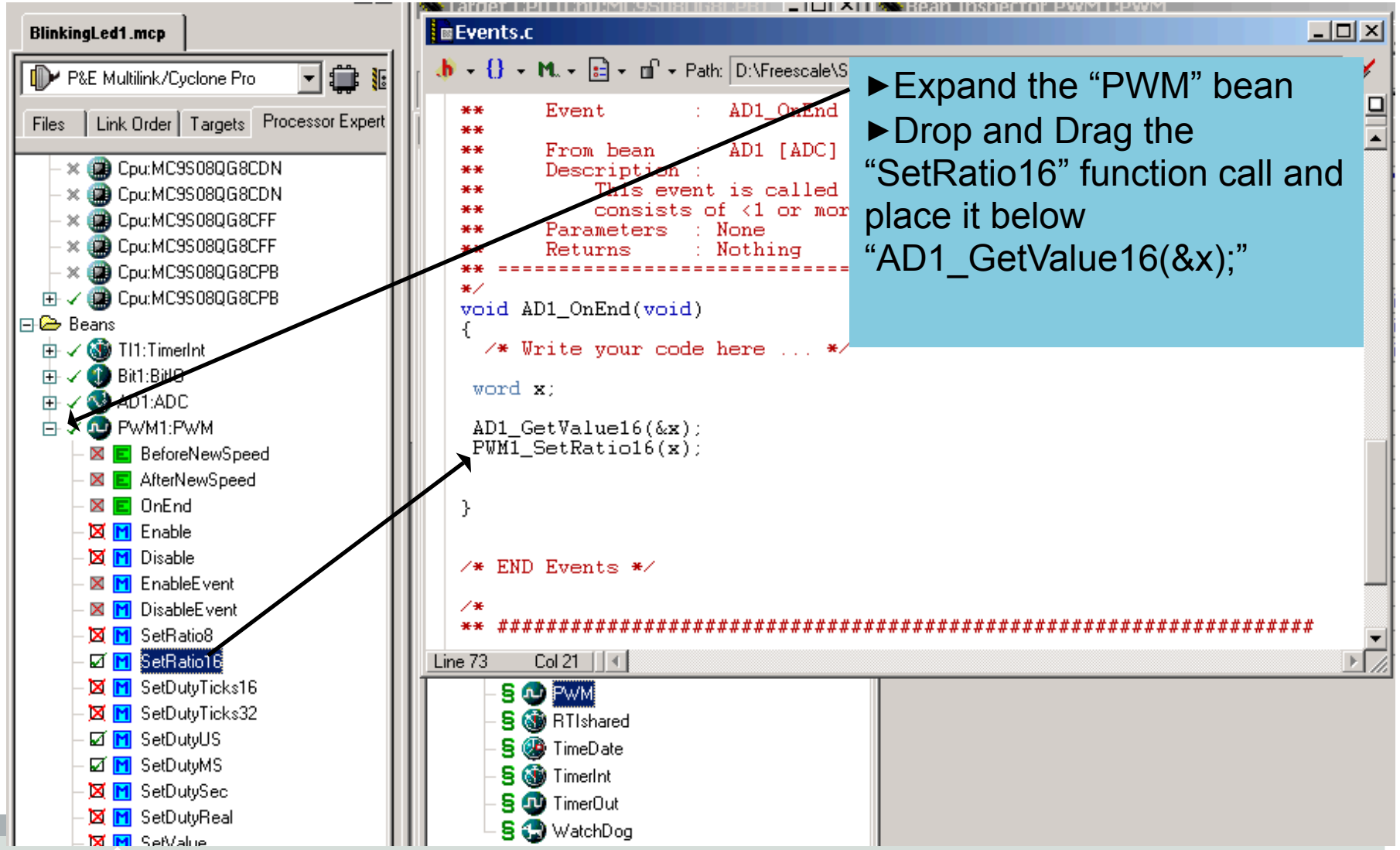
freescale semiconductor

LAB3 – ADC & PWM



freescale semiconductor

LAB3 – ADC & PWM

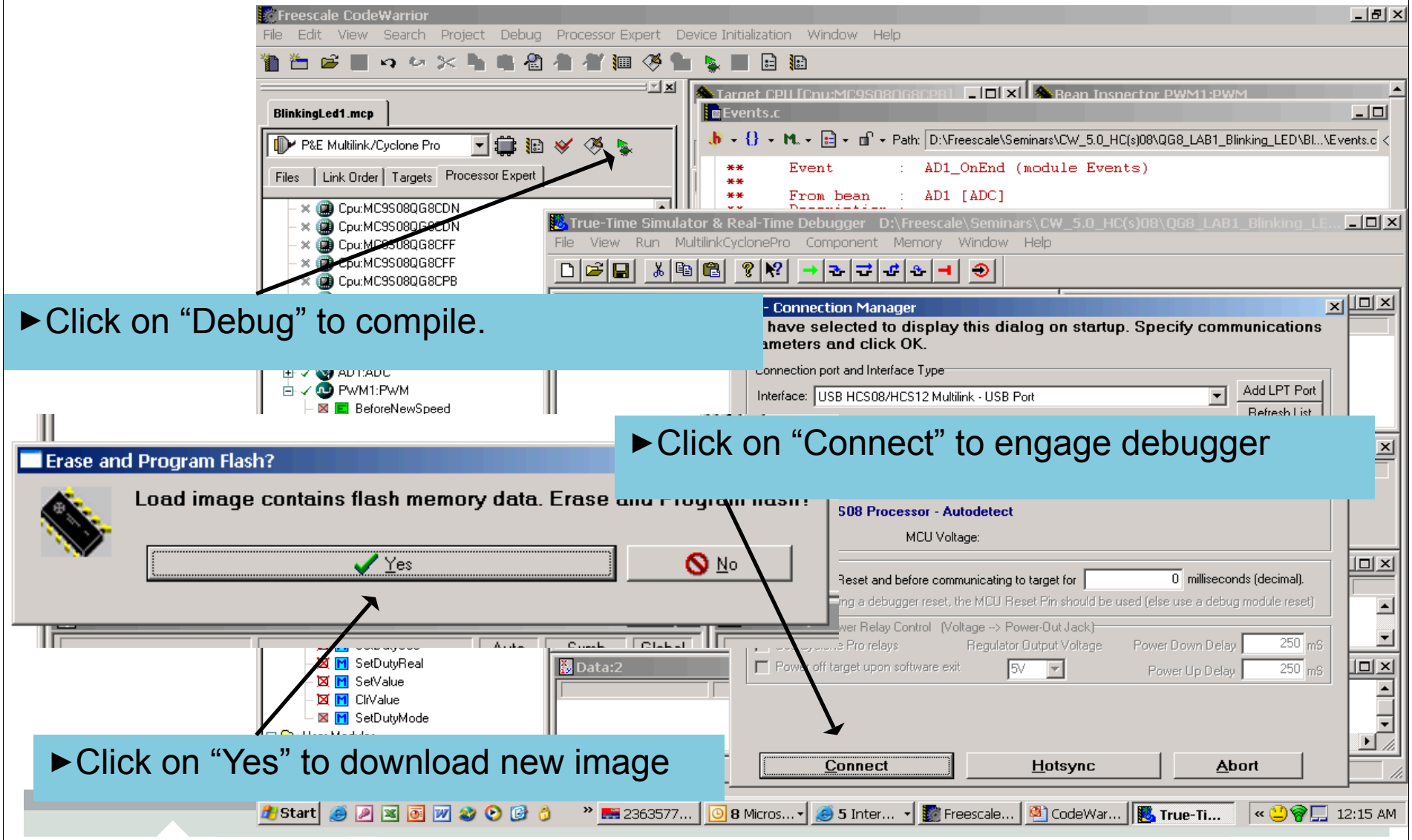


►Expand the “PWM” bean

►Drop and Drag the “SetRatio16” function call and place it below “AD1_GetValue16(&x);”

freescale semiconductor

LAB3 – ADC & PWM



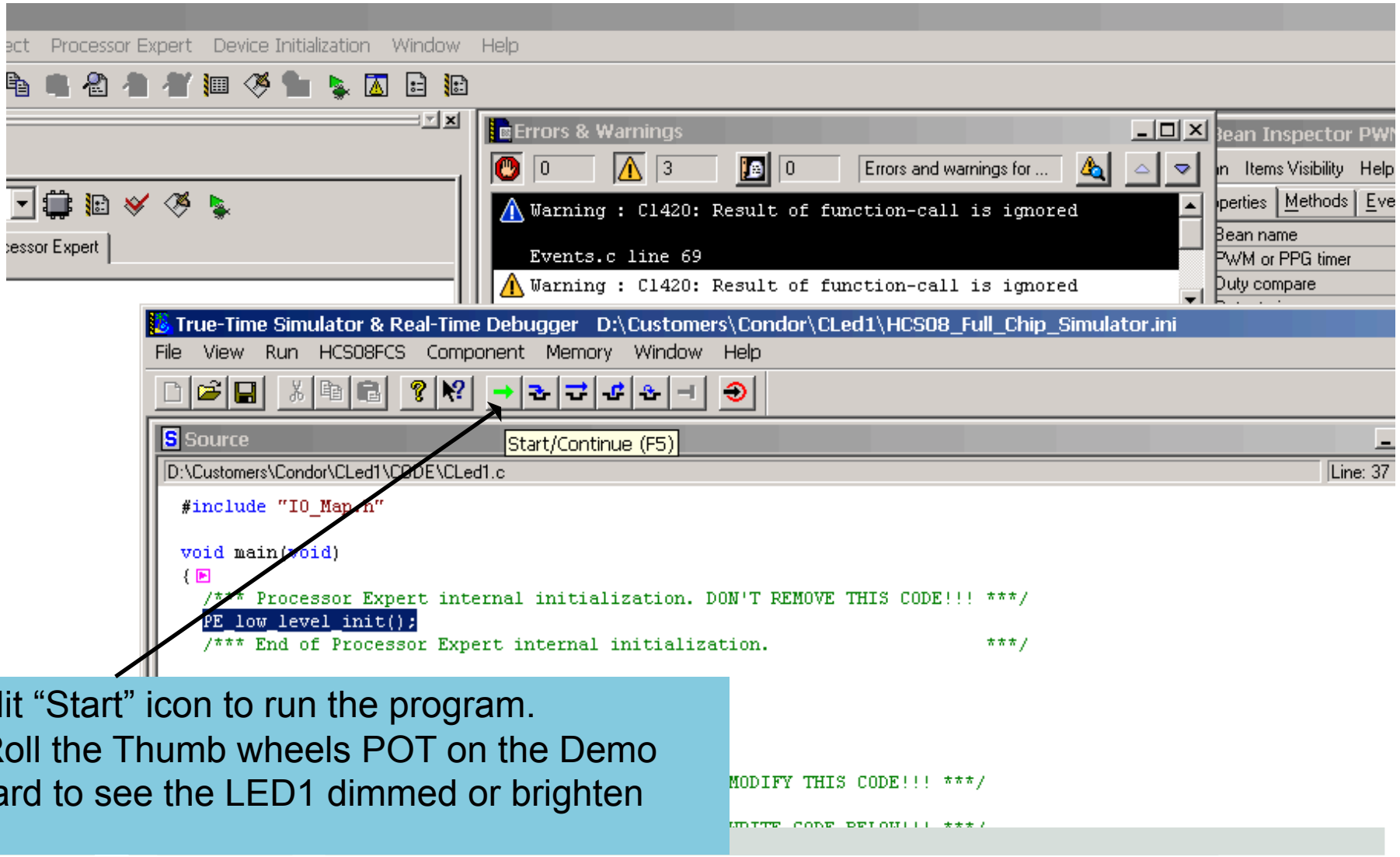
►Click on “Debug” to compile.

►Click on “Connect” to engage debugger

►Click on “Yes” to download new image

freescale semiconductor

LAB3 – ADC & PWM



►Hit “Start” icon to run the program.
►Roll the Thumb wheels POT on the Demo board to see the LED1 dimmed or brighten

freescale semiconductor

LAB4 – SCI Communication

Create a brand new project and set bus clk to 8Mhz.

Initialize SCI(Serial) Comm to 9600 Baud rate

Set Interrupt upon received of Character.

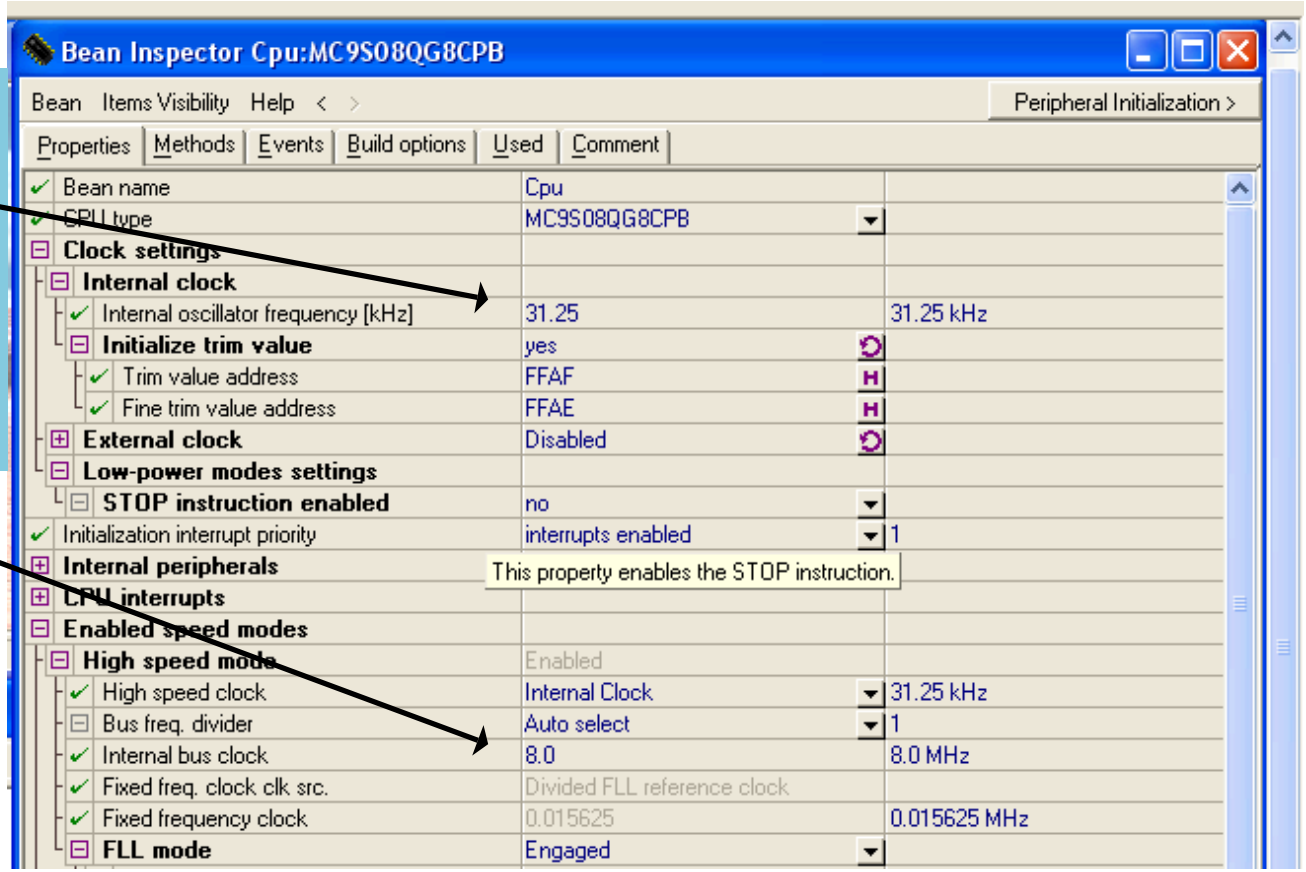
Set Hyperterminal on PC to 9600 baud.

Write application to echo received character back to PC

LAB4 – SCI Communication

►Change the Internal Osc Frequency to 31.25Khz.

►Change the Internal BusClock to 8.0 Mhz



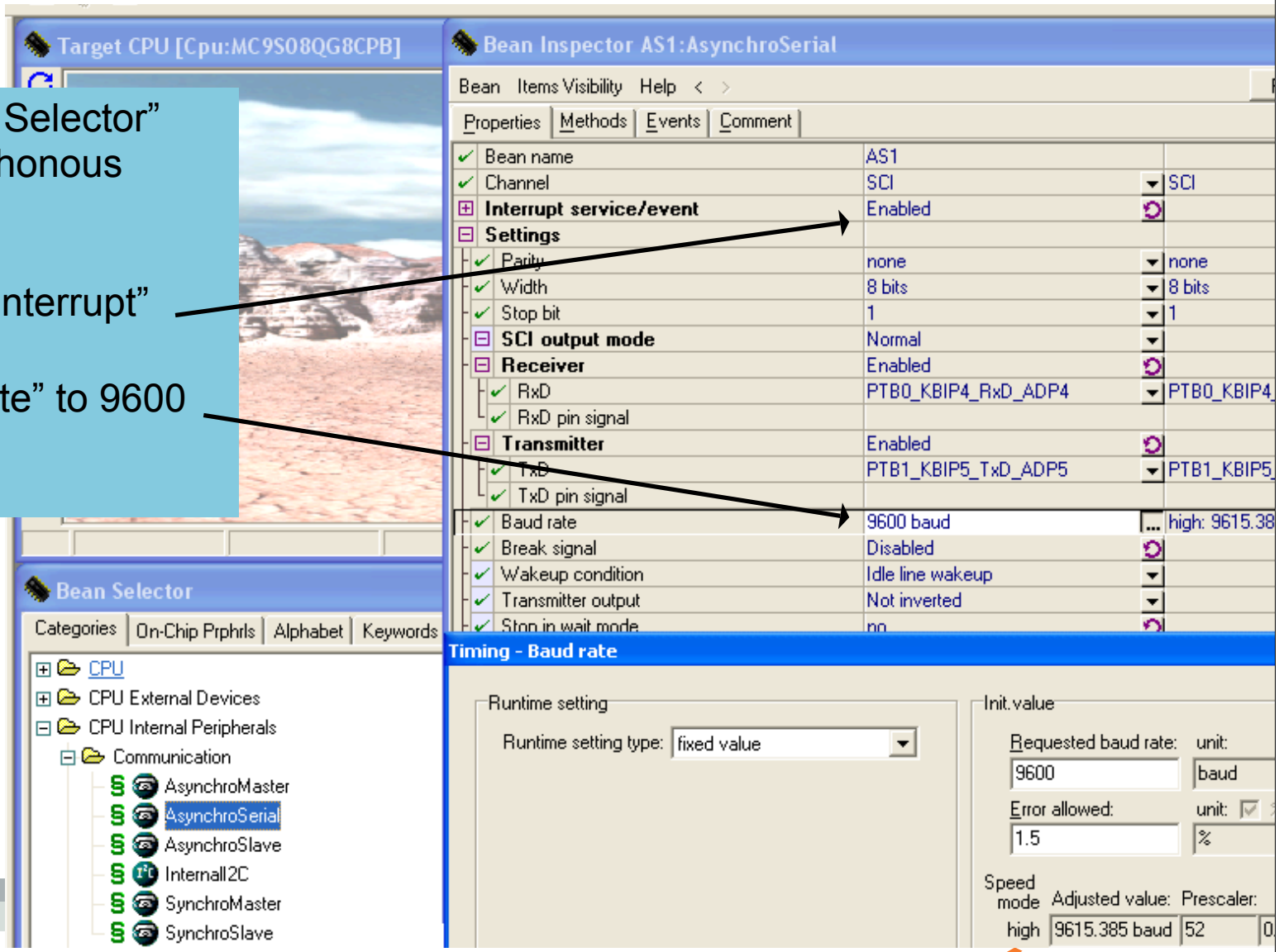
freescale semiconductor

LAB4 – SCI Communication

►In the “Bean Selector” select “Asyncrhonous Serial” bean.

►Enable the “Interrupt”

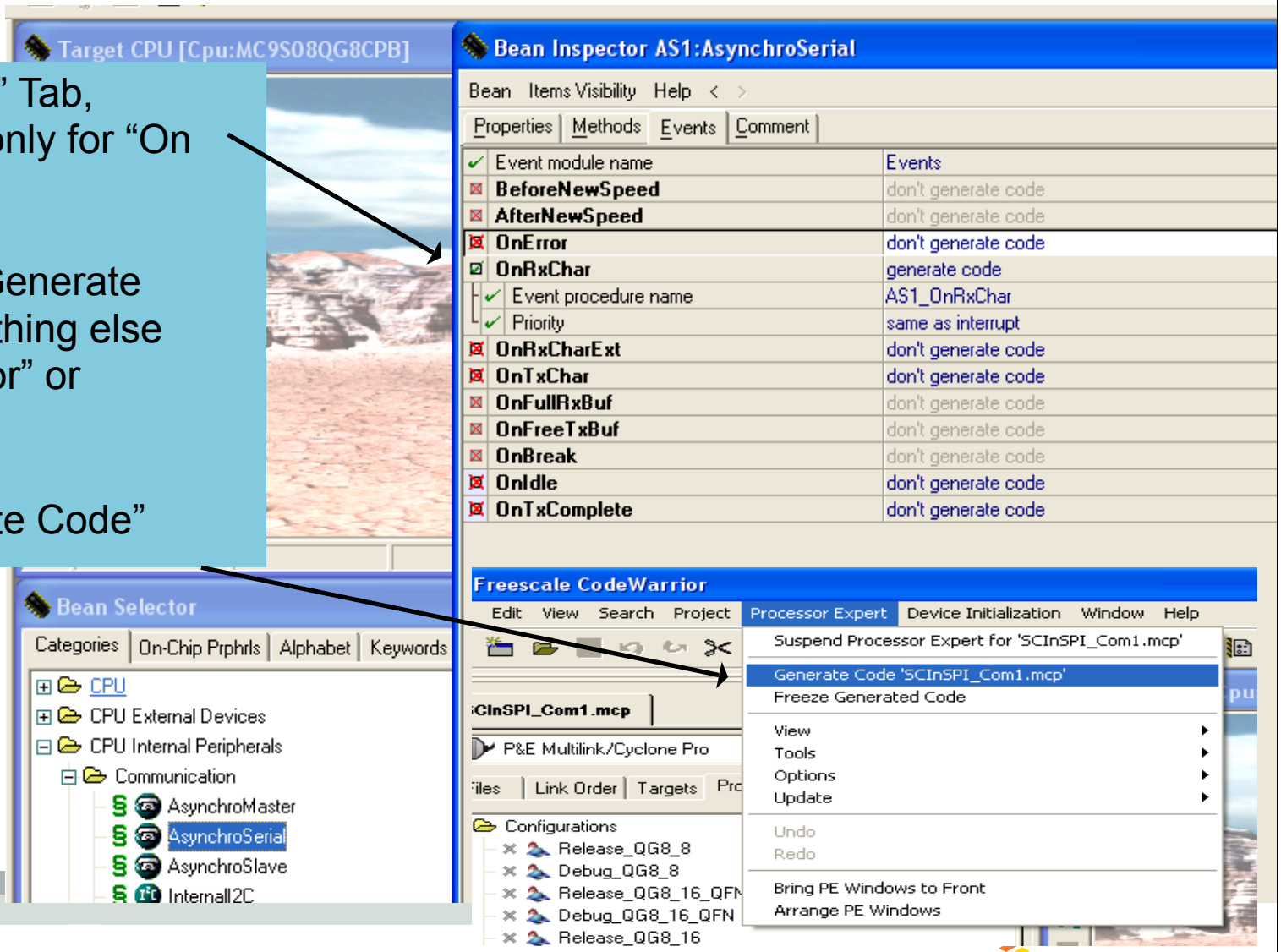
►Set “Baud rate” to 9600



freescale semiconductor

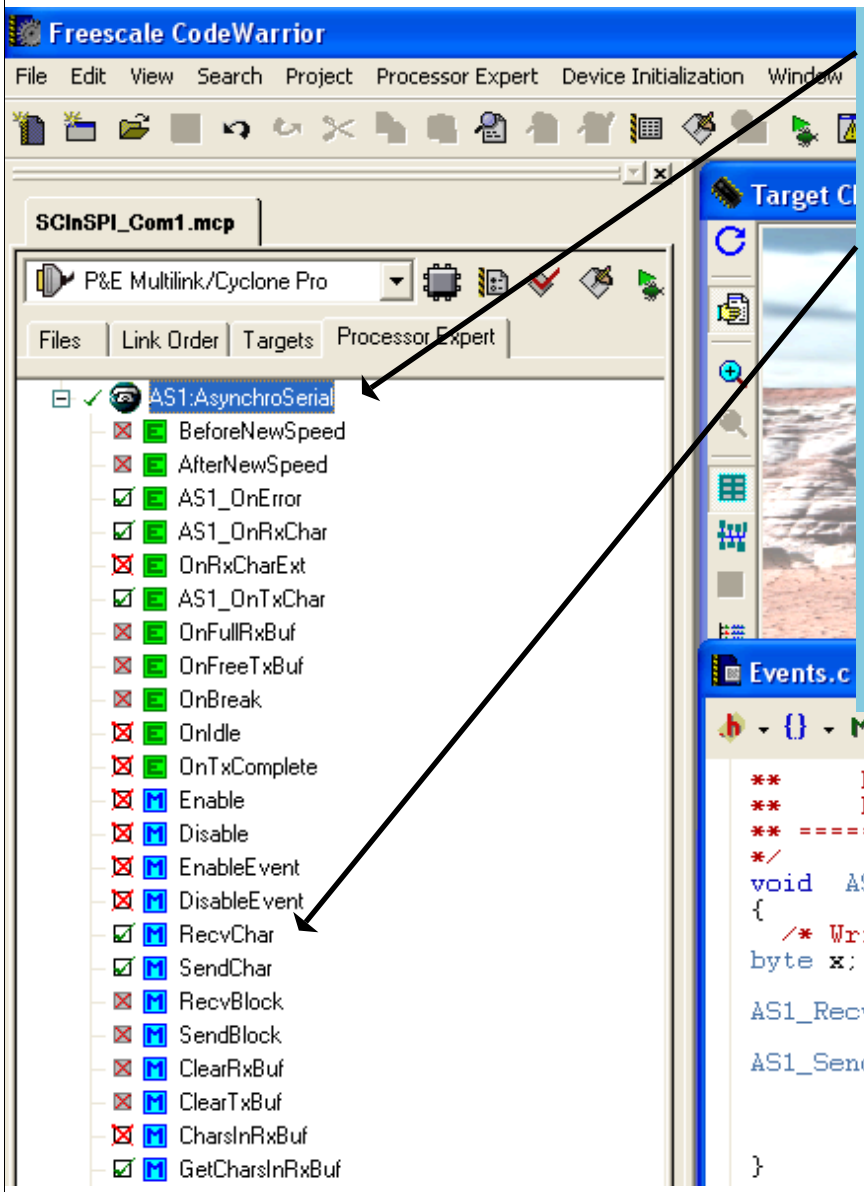
LAB4 – SCI Communication

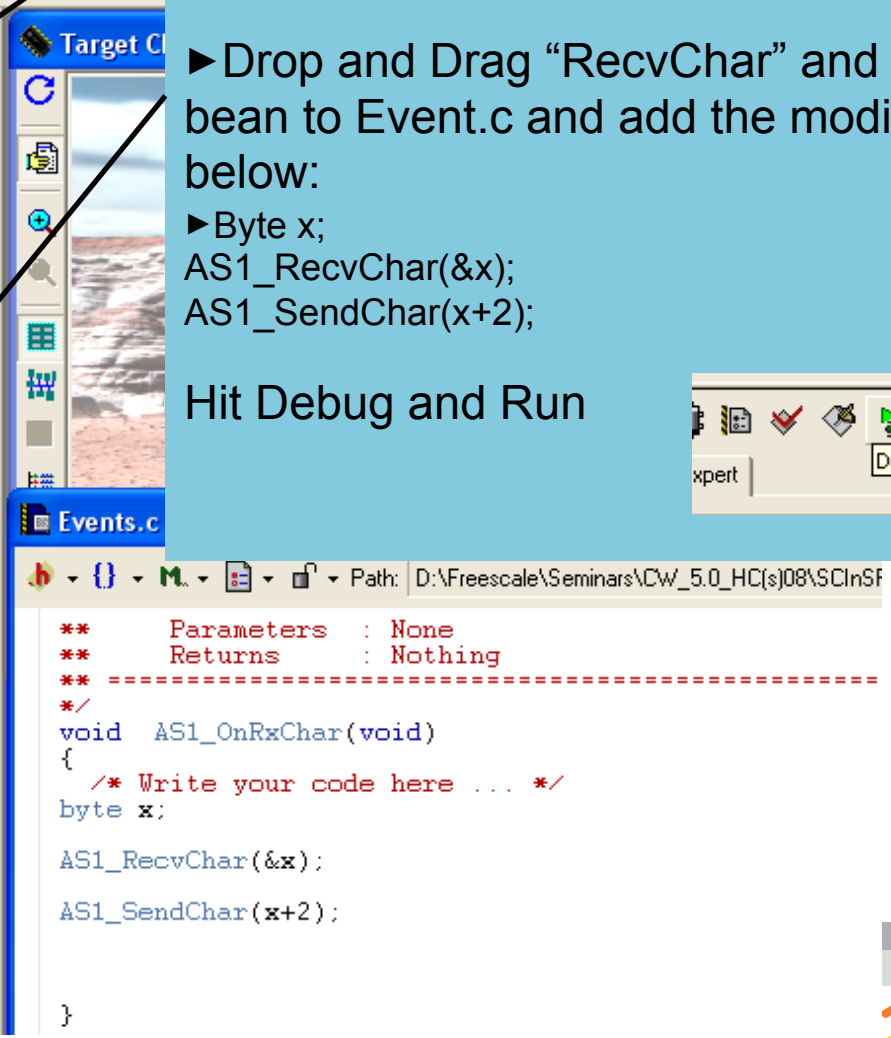
- In the “Events” Tab, generate code only for “On RXChar”
- Select Don’t Generate Code” for everything else such as “OnError” or “OnTxChar”
- Click “Generate Code”



freescale semiconductor

LAB4 – SCI Communication





►Under “Processor Expert” Tab, expand the “ AS1 AsynchronousSerial” bean

►Drop and Drag “RecvChar” and “SendChar” bean to Event.c and add the modification below:

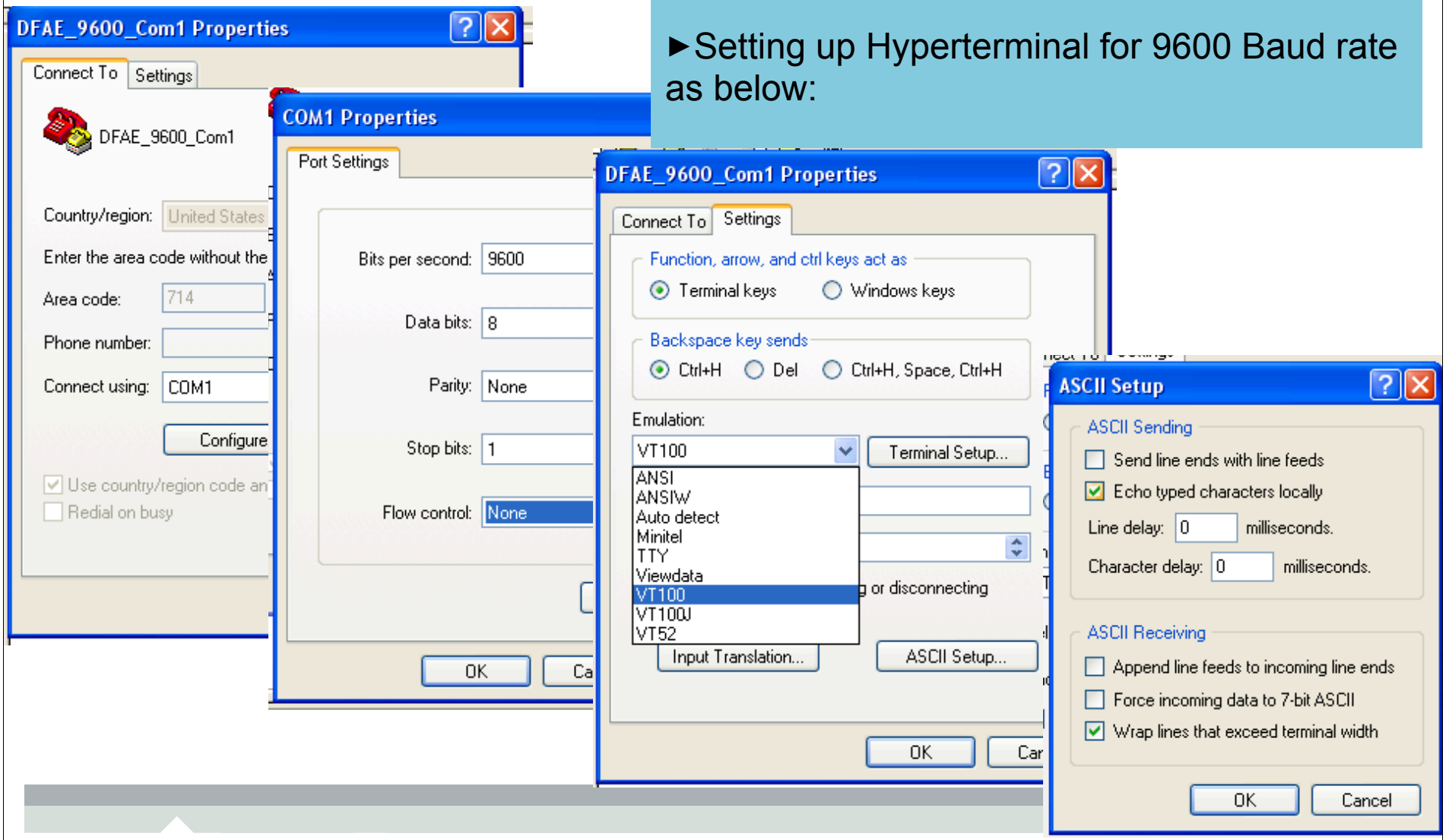
►Byte x;
AS1_RecvChar(&x);
AS1_SendChar(x+2);

Hit Debug and Run

freescale semiconductor

LAB4 – SCI Communication

►Setting up Hyperterminal for 9600 Baud rate as below:



freescale semiconductor

LAB5 – SPI Communication

- To send communication from PC to PC via an QG8 SCI/SPI converter
- Continuation with LAB4 (SCI Comm).
- Initialize SPI for MasterMode
- Initialize SPI for Slave Mode
- Write SCI/SPI Converter code
- Use Hyperterminal to test and application.

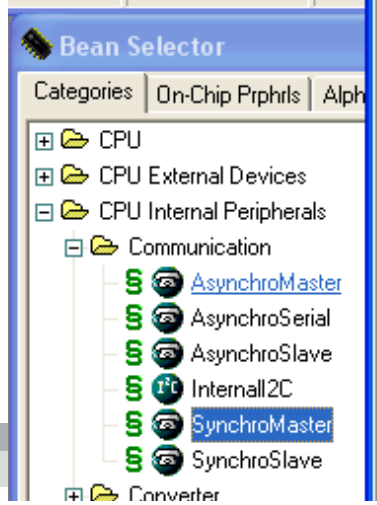
LAB5 – SPI Communication(Master)

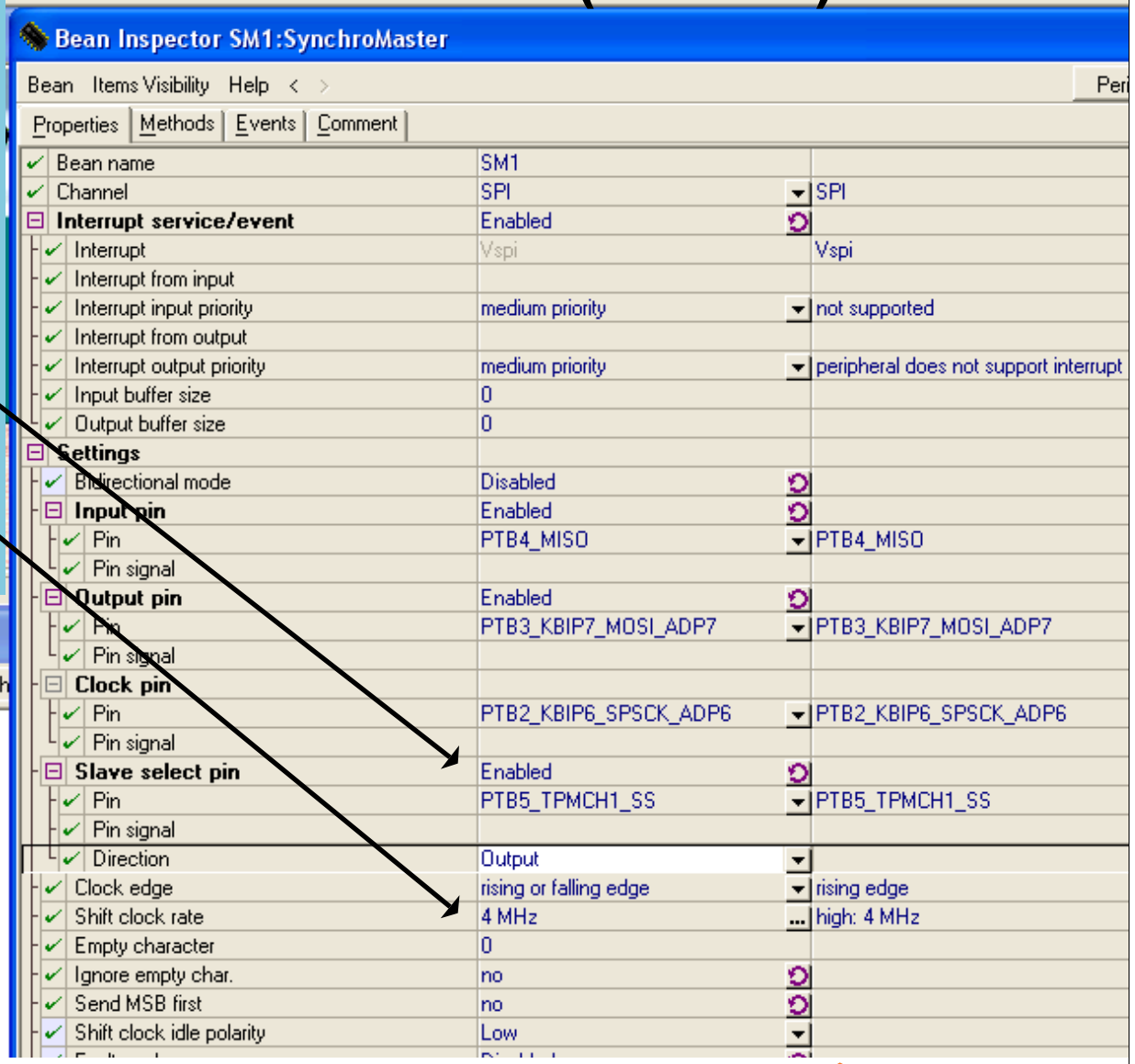
SPI Master Configuration

►In the “Bean Selector” window, select Synchronous Master.

►Enable “Slave Select pin” and set Direction to Output

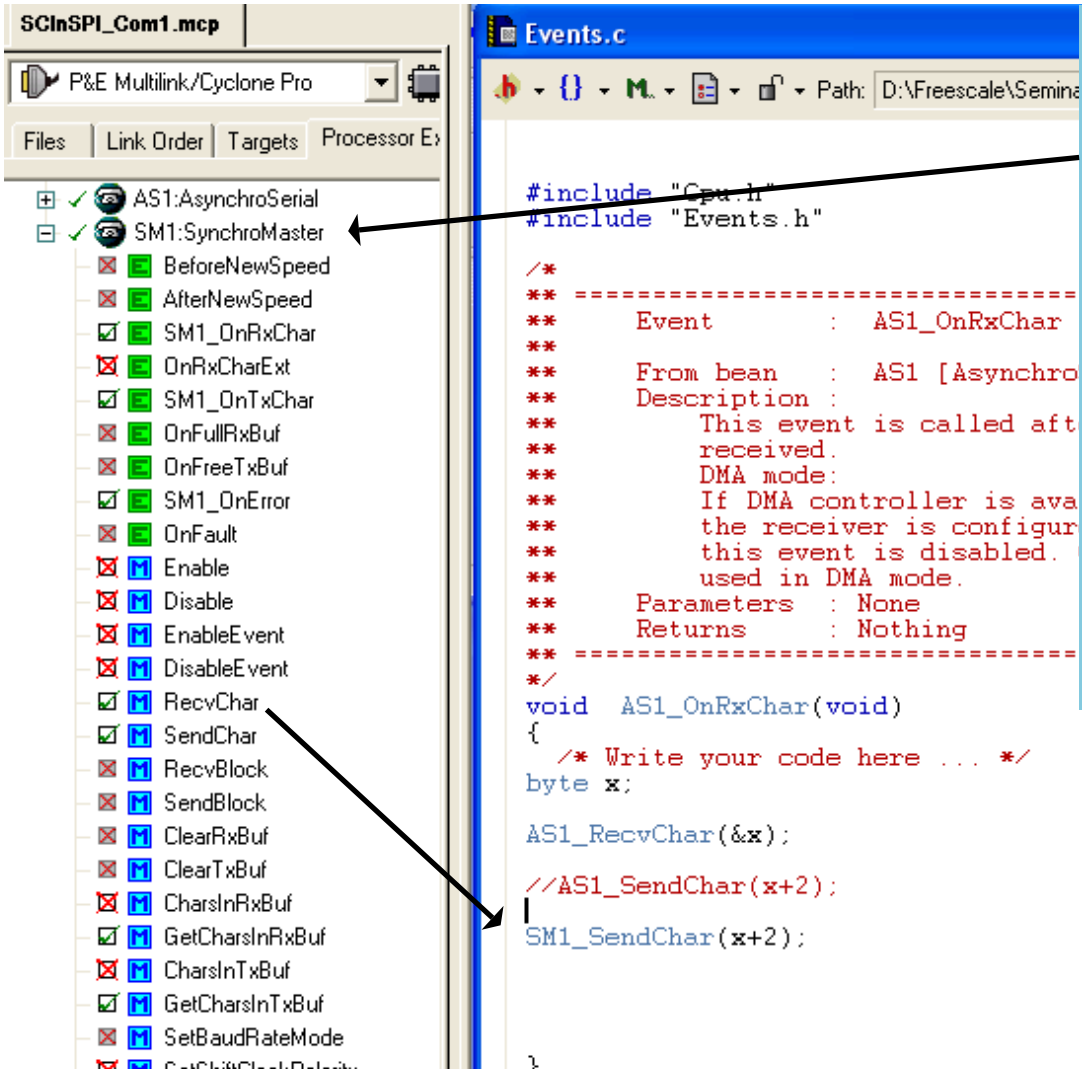
►Set “Clock rate” to 4Mhz





freescale semiconductor

LAB5 – SPI Communication(Master)



►Under “Processor Expert” Tab, expand the “SM1:SyncrhronousMaster” bean

►Drop and Drag “SendChar” and modifiy as below:

SM1_SendChar(x+2);

►Hit Debug and Run

►Use Hyper-terminal to test Communication

freescale semiconductor

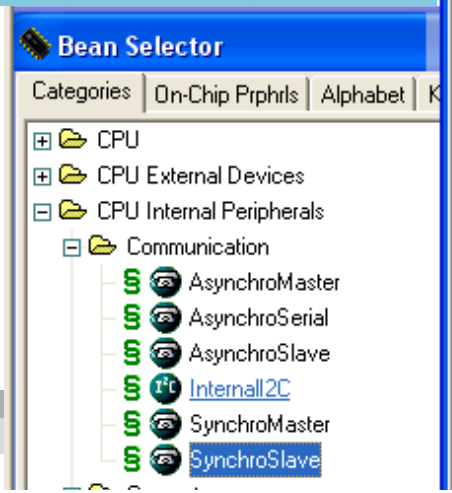
LAB5 – SPI Communication(Slave)

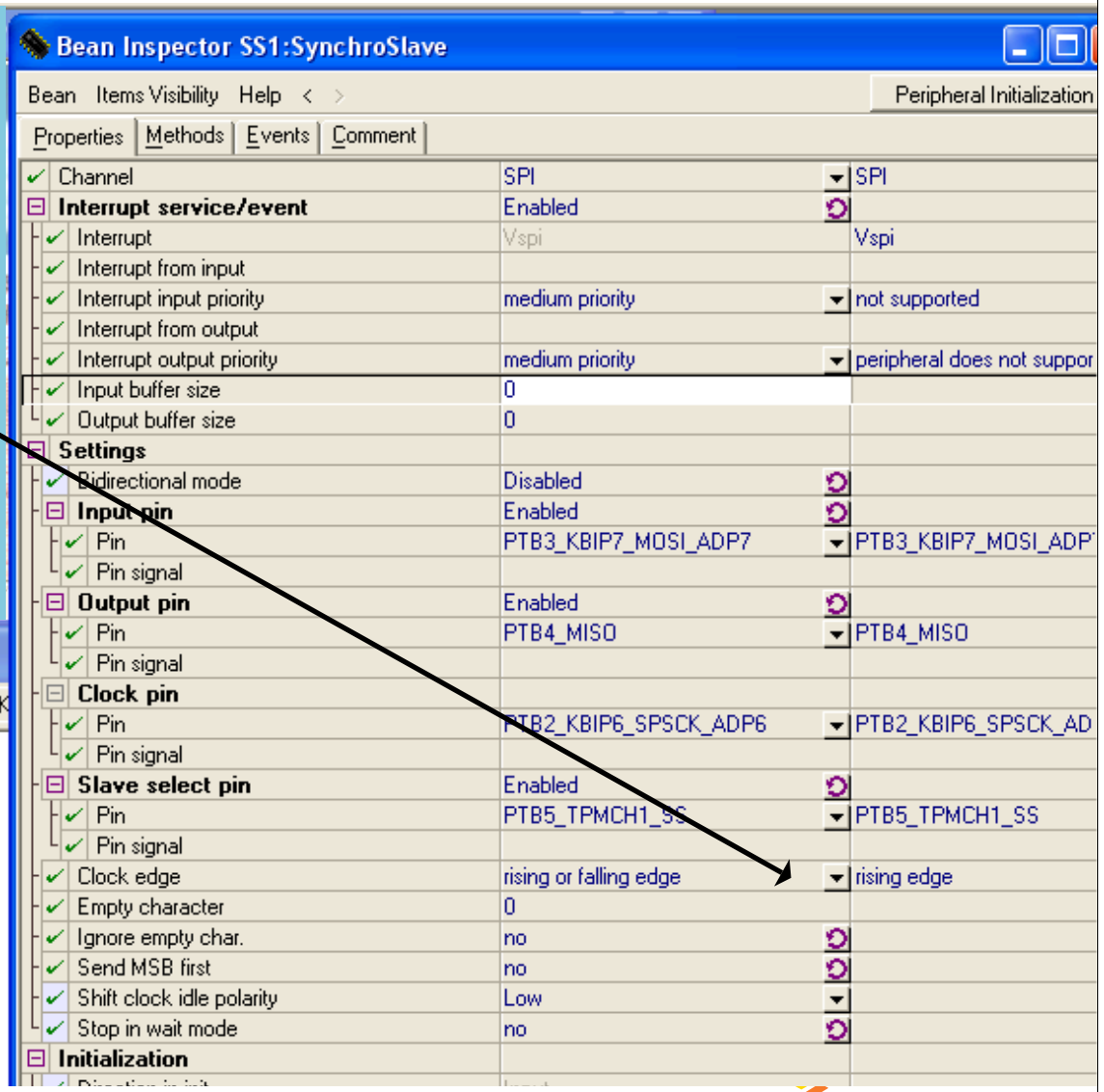
SPI Slave Configuration

►In the “Bean Selector” window, select SynchronousSlave.

►Set “Clock Edge” to rising edge to match up with the Master mode.

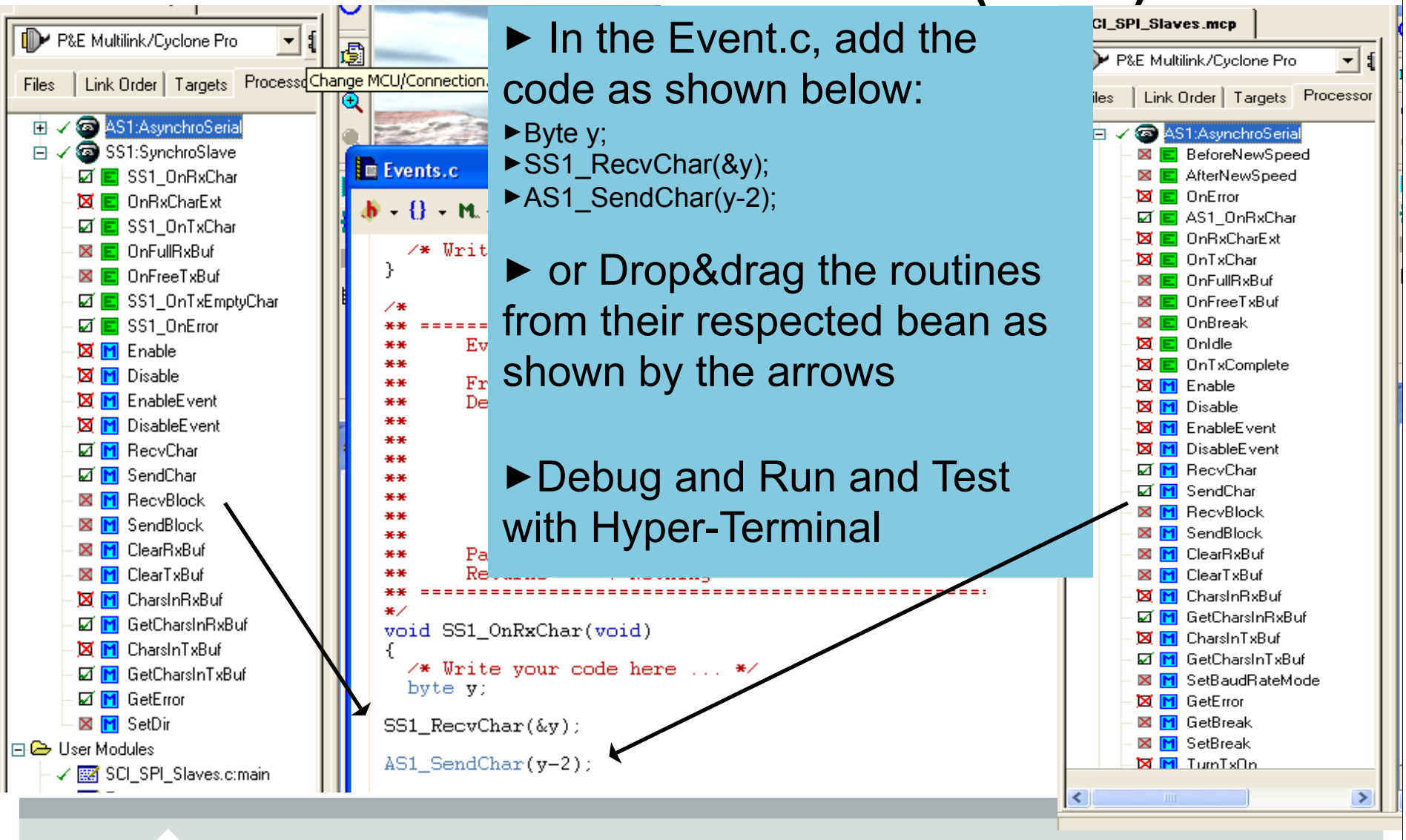
►Generate Code





freescale semiconductor

LAB5 – SPI Communication(Slave)



► In the Event.c, add the code as shown below:

►Byte y;

►SS1_RecvChar(&y);

►AS1_SendChar(y-2);

► or Drop&drag the routines from their respected bean as shown by the arrows

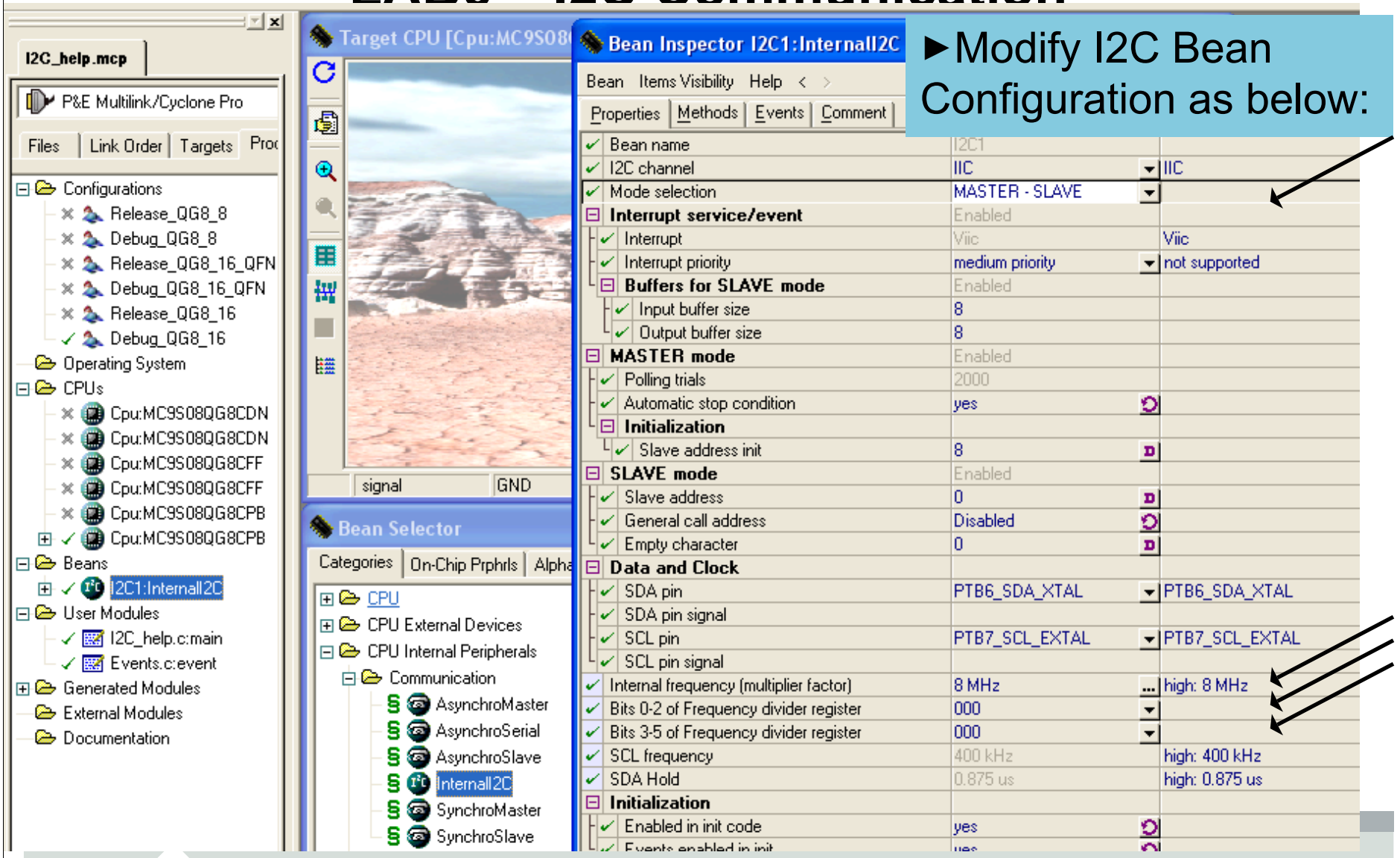
►Debug and Run and Test with Hyper-Terminal

freescale semiconductor

LAB6 – I2C Communication

- Modify HW to add 4.7K resistors to SCL and SDA signals
- Initialize I2C HW
- Write I2C application

LAB6 – I2C Communication



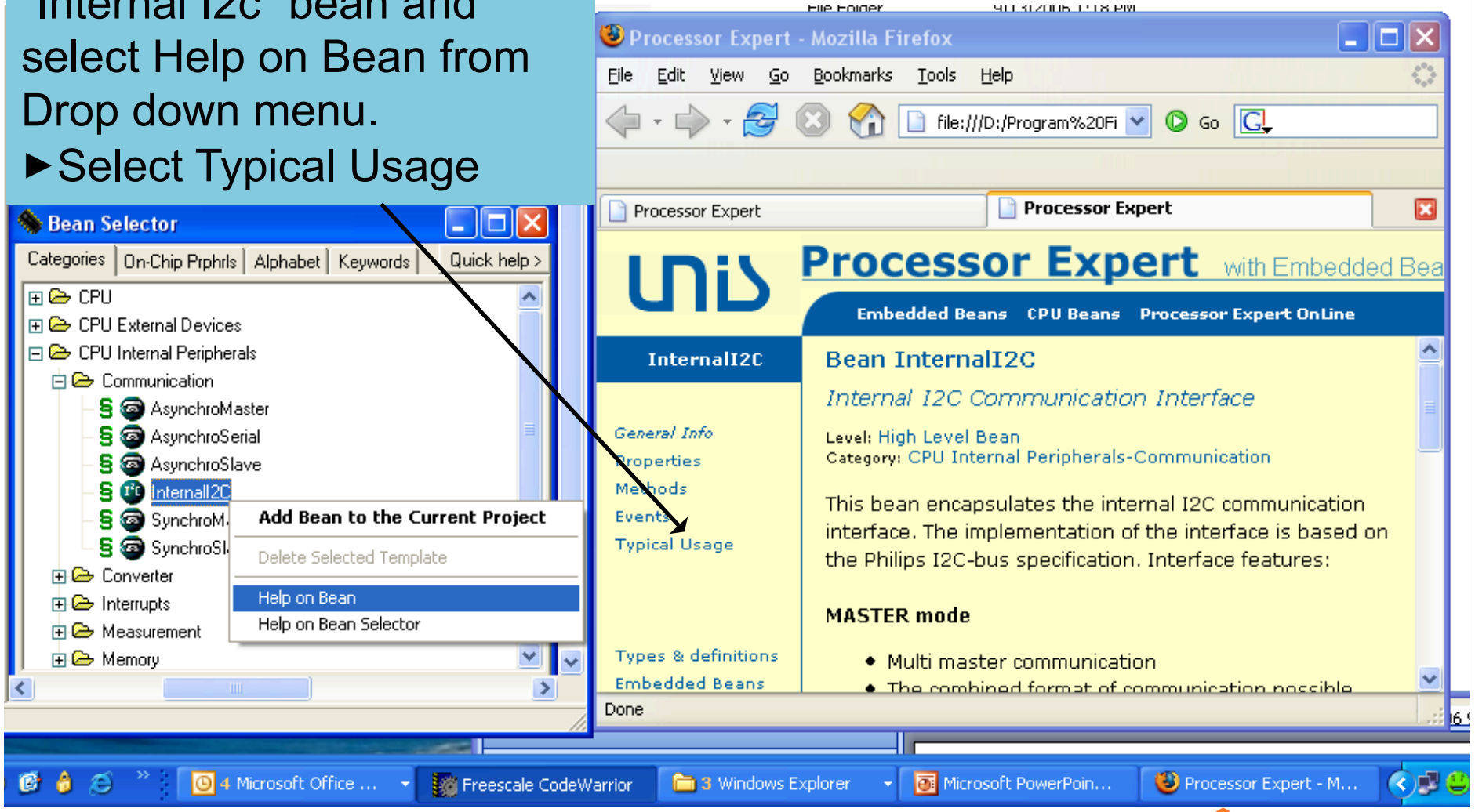
►Modify I2C Bean Configuration as below:

freescale semiconductor

LAB6 – I2C Communication

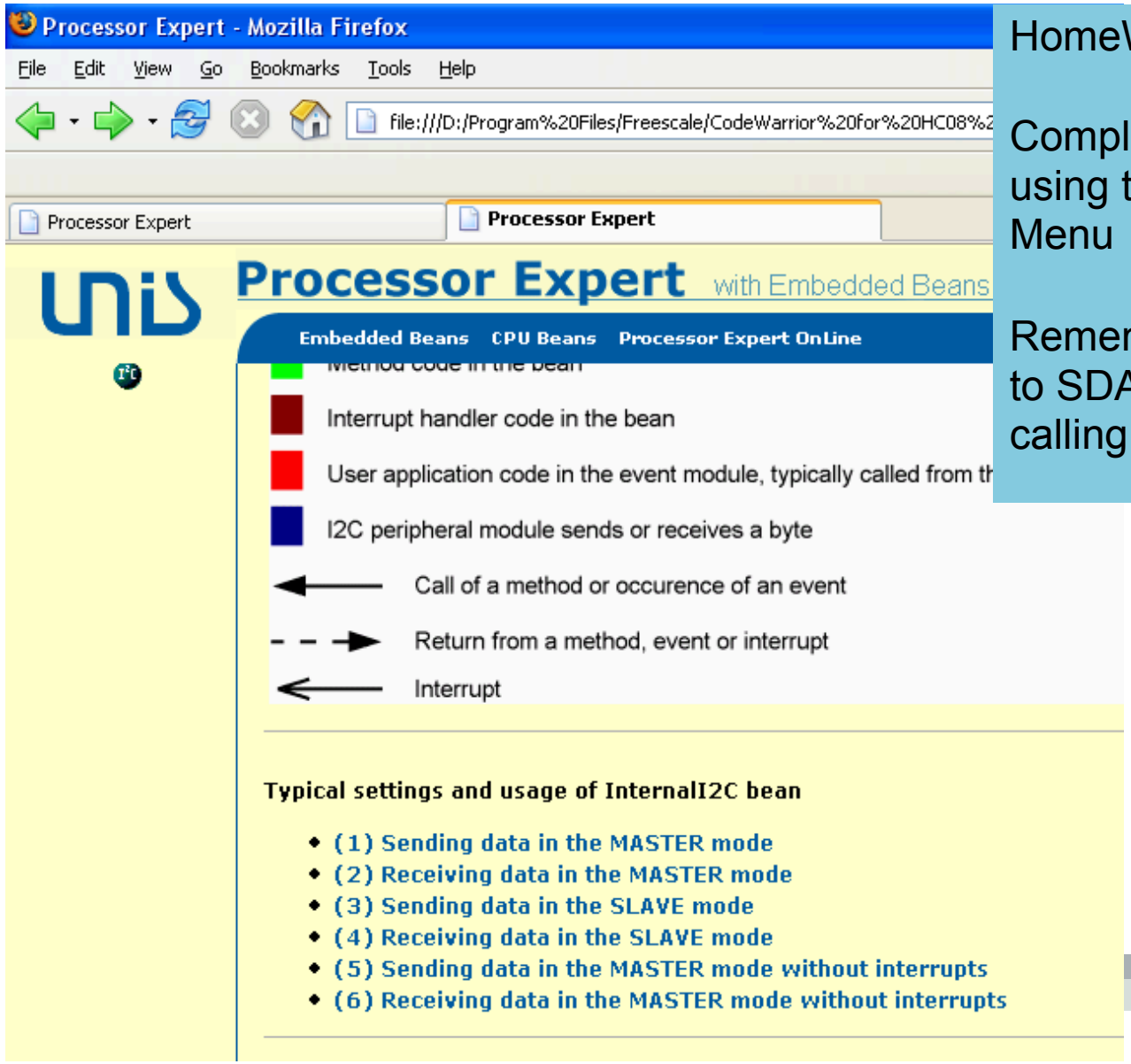
►Right hand click on "Internal I2c" bean and select Help on Bean from Drop down menu.

►Select Typical Usage



freescale semiconductor

LAB6 – I2C Communication



HomeWork:

Complete an I2C application using the Typical Usage Help Menu

Remember to add 4.7k resistors to SDA and SCL line before calling He My FAE.

freescale semiconductor

Thank You