XC2287M HOT

Solution ADC_CC2

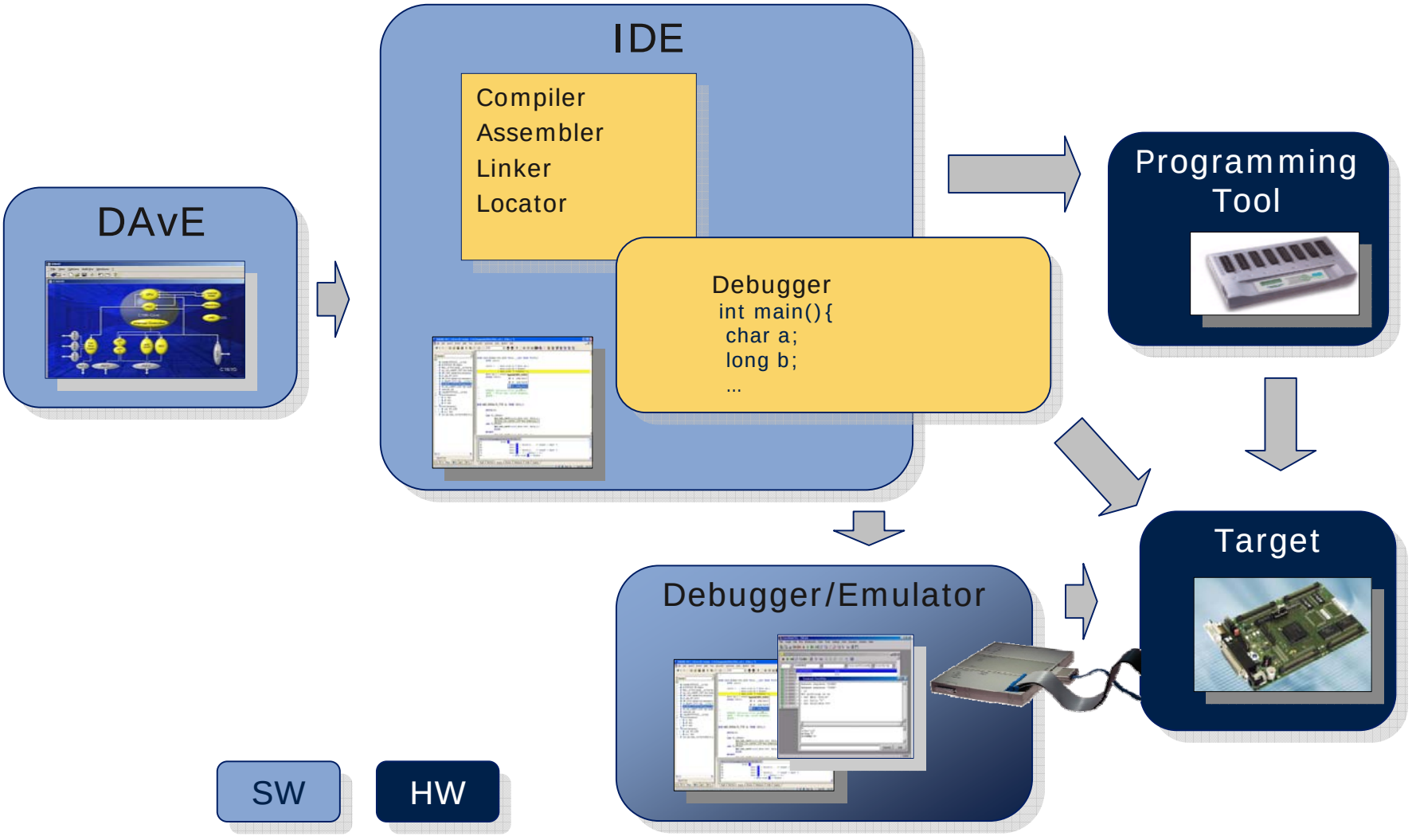
Flashing LED’s by using the ADC Interrupt with CAPCOM_2

Device: XC2287M-104F80
Compiler: Tasking Viper 2.4r1
Code Generator: DAvE 2.1

Never stop thinking

Hardware Triggering of ADC

XC2287M HOT Exercise ADC_CC2
Interaction of Development Tools



HOT Exercise ADC_CC2
Hardware Triggering of ADC

In this exercise you will:

- Configure the XC2287M with DAvE
- Assign a potentiometer to an ADC cannel
- Define ADC parameters
- Develop a program that will trigger ADC conversion by hardware

HOT Exercise ADC_CC2
Hardware Triggering of ADC

Exercise goal:

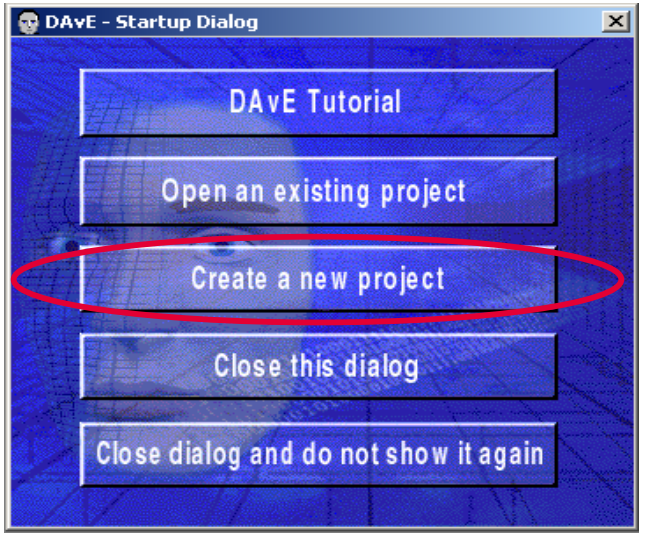
Trigger ADC conversion by capture / compare of CC2_17

- The onboard potentiometer is connected to channel 0 (P5.0)
- A/D converter will run in hardware triggered mode
- Use CC2_17 compare match to trigger ADC conversion
- Output A/D results on Port P10 (LED display)

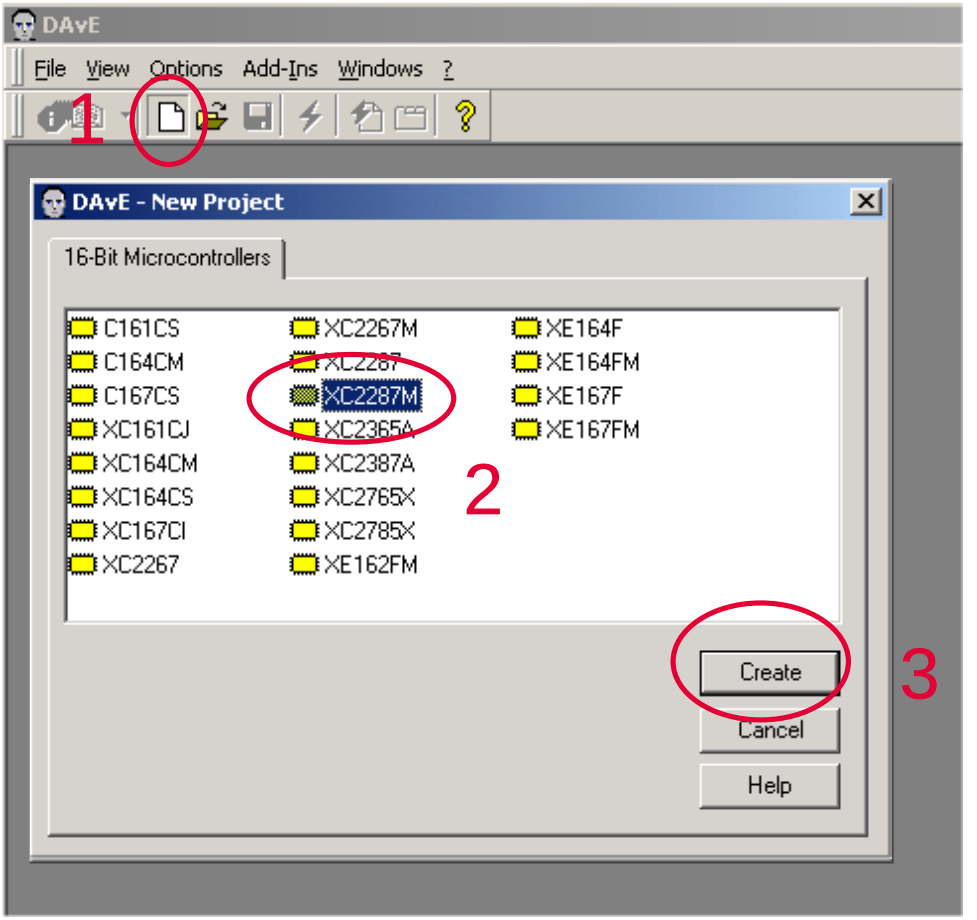
HOT Exercise ADC_CC2
Start DAvE

- **Start DAvE**
  - Click on the 

- **Create a new project** (Startup Dialog pop up automatically)
  - Click on ‘Create a new project’ or select File -> New
  - Select microcontroller: ‘XC2287M’

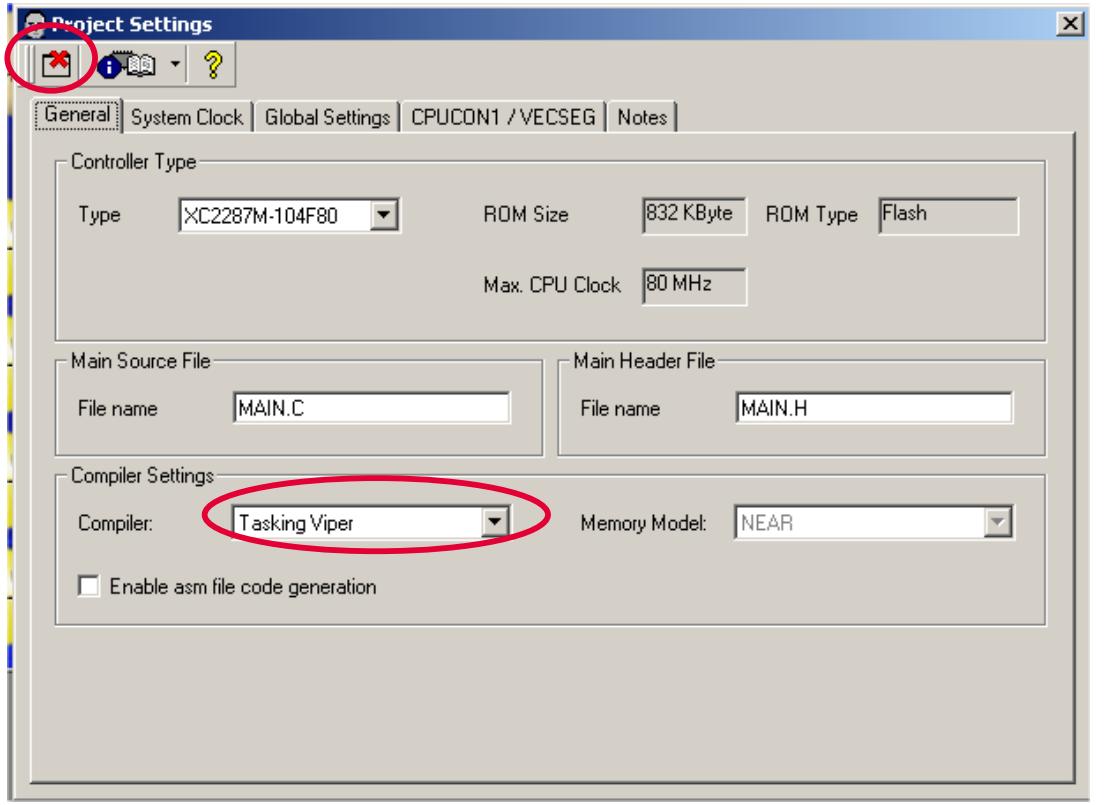


HOT Exercise ADC_CC2
Select Device



HOT Exercise ADC_CC2 - DAvE Configurations
Project Settings

- Project Settings
- Close the window

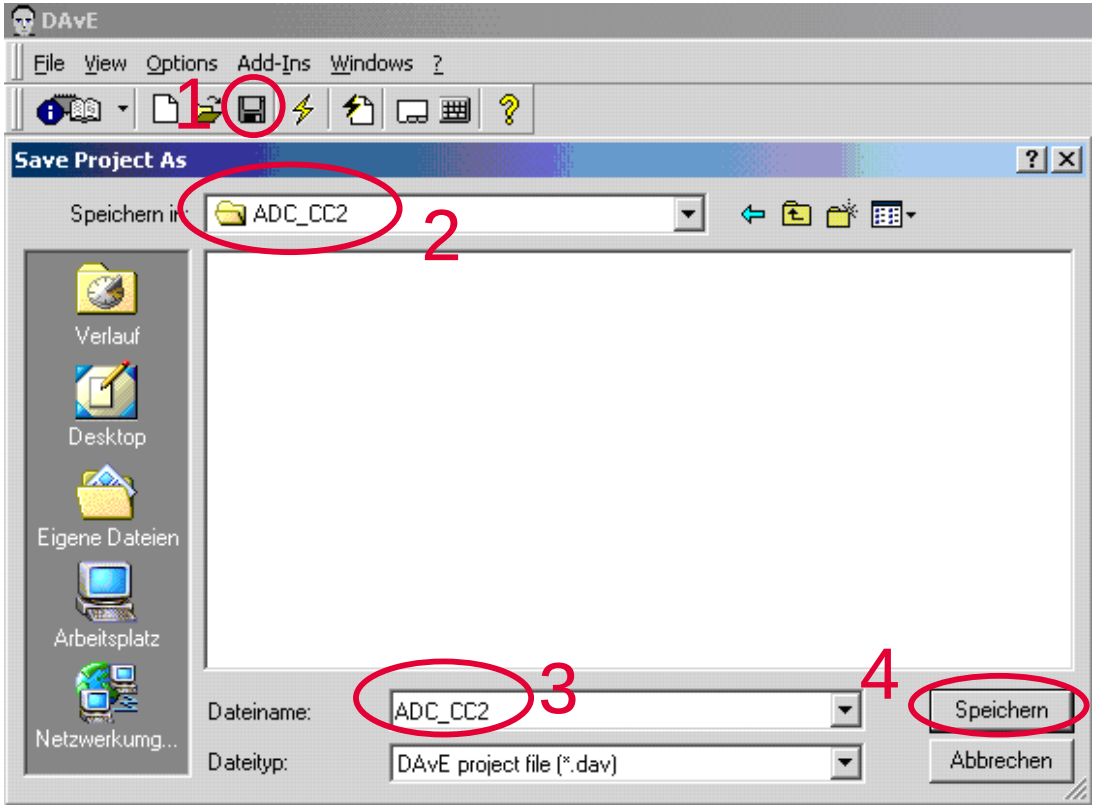


HOT Exercise ADC_CC2 - DAvE Configurations
Save DAvE Project

- Save your DAvE project 
  - Path:
    C:\IFX_HOT\XC2287M\Examples\ADC_CC2
  - Project name:
    ADC_CC2\ADC_CC2.dav

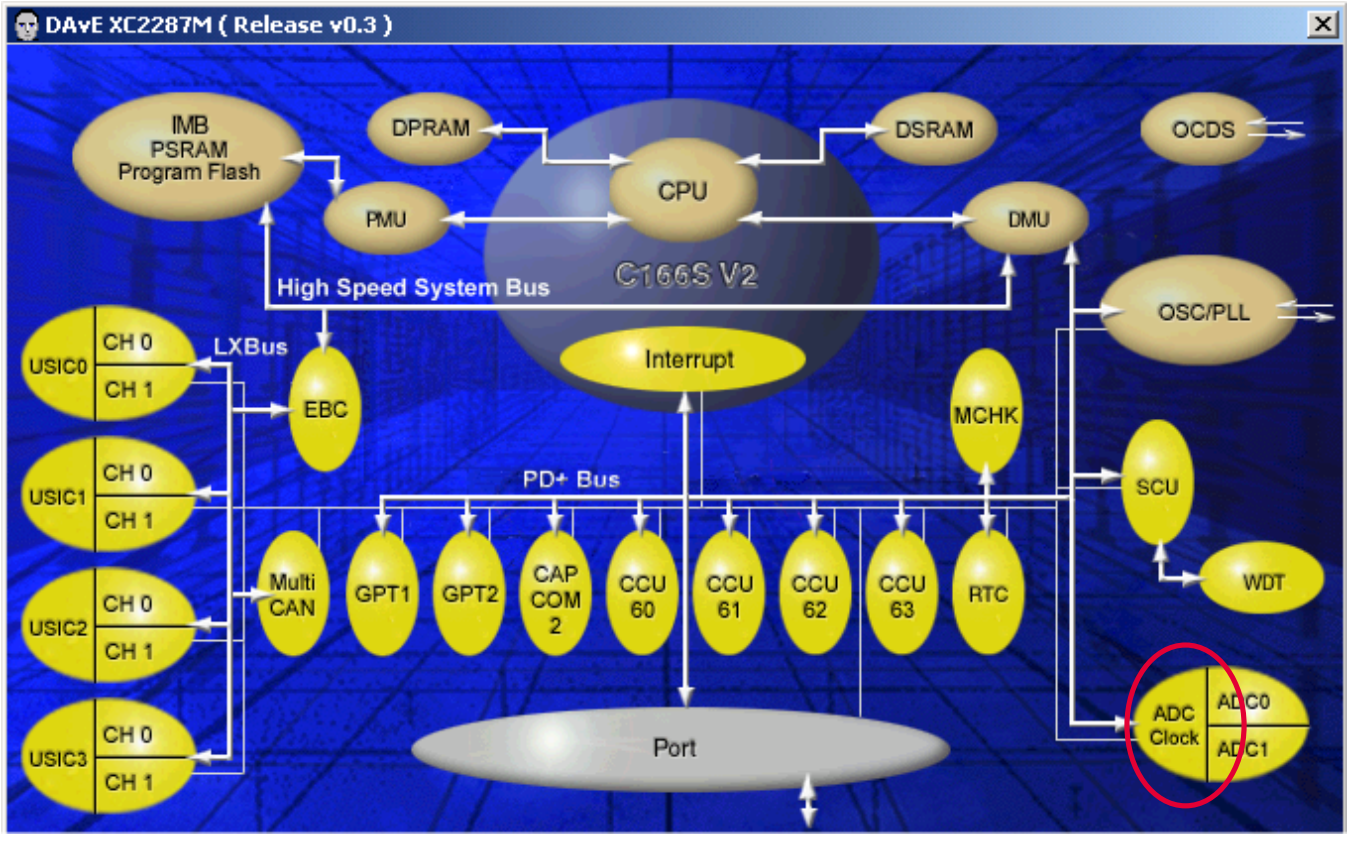
HOT Exercise ADC_CC2 - DAvE Configurations
Save DAvE Project

- Save your DAvE Project File

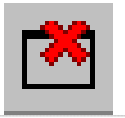


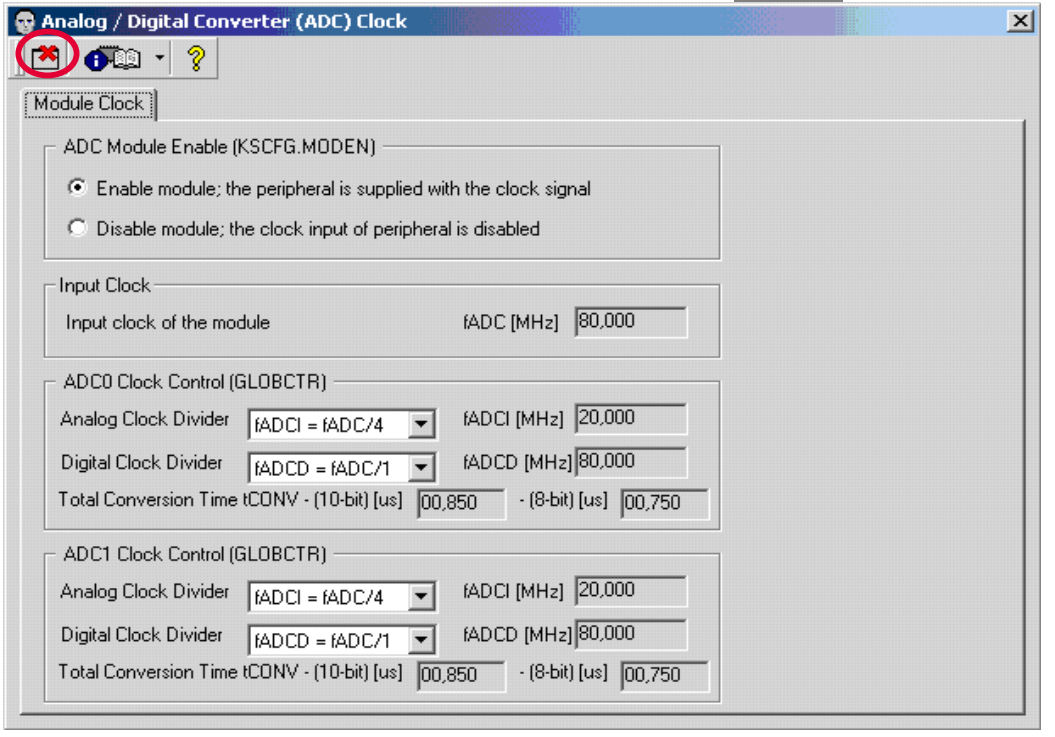
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings

- Click on ‘ADC Clock’ 



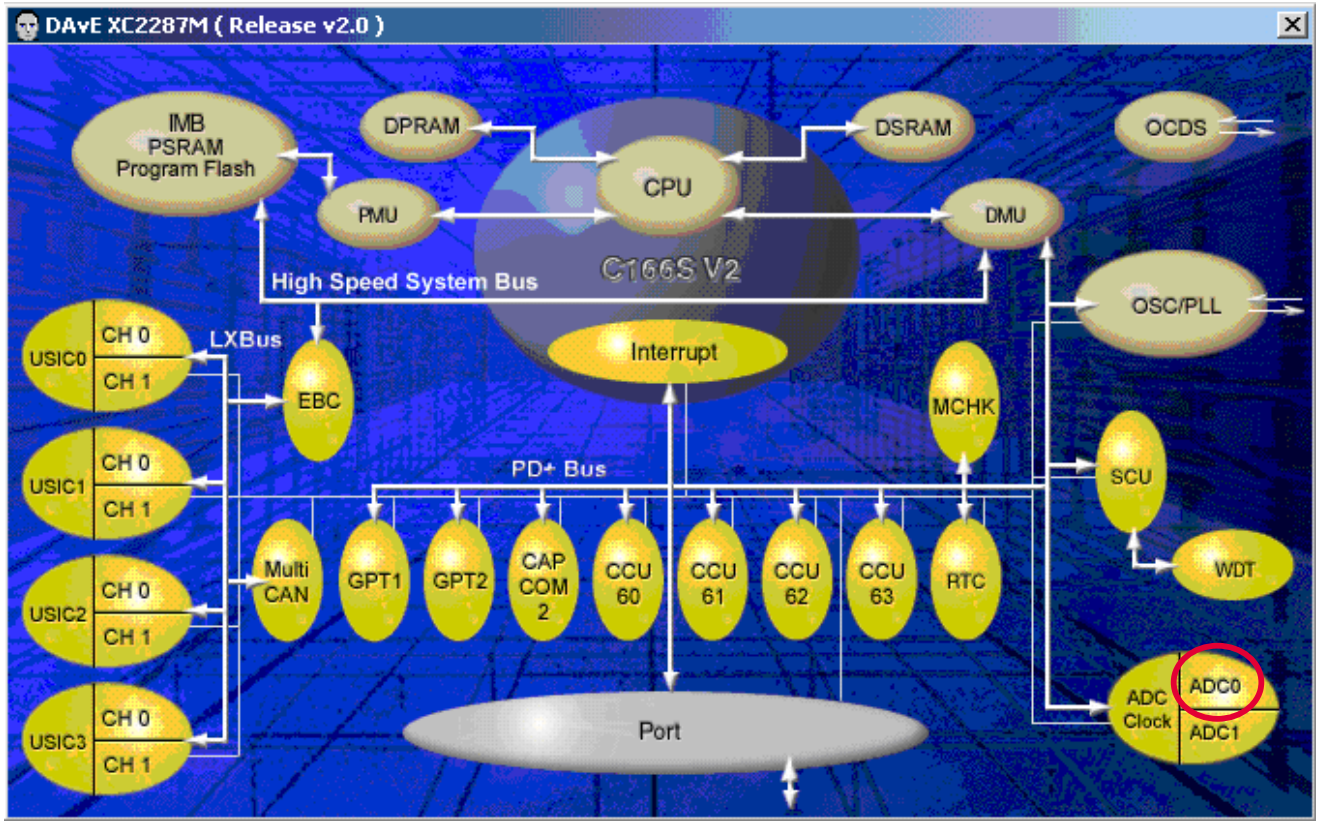
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ‘ADC Clock’
  - Enable module
  - default settings for others
  - Close the windows by pressing 



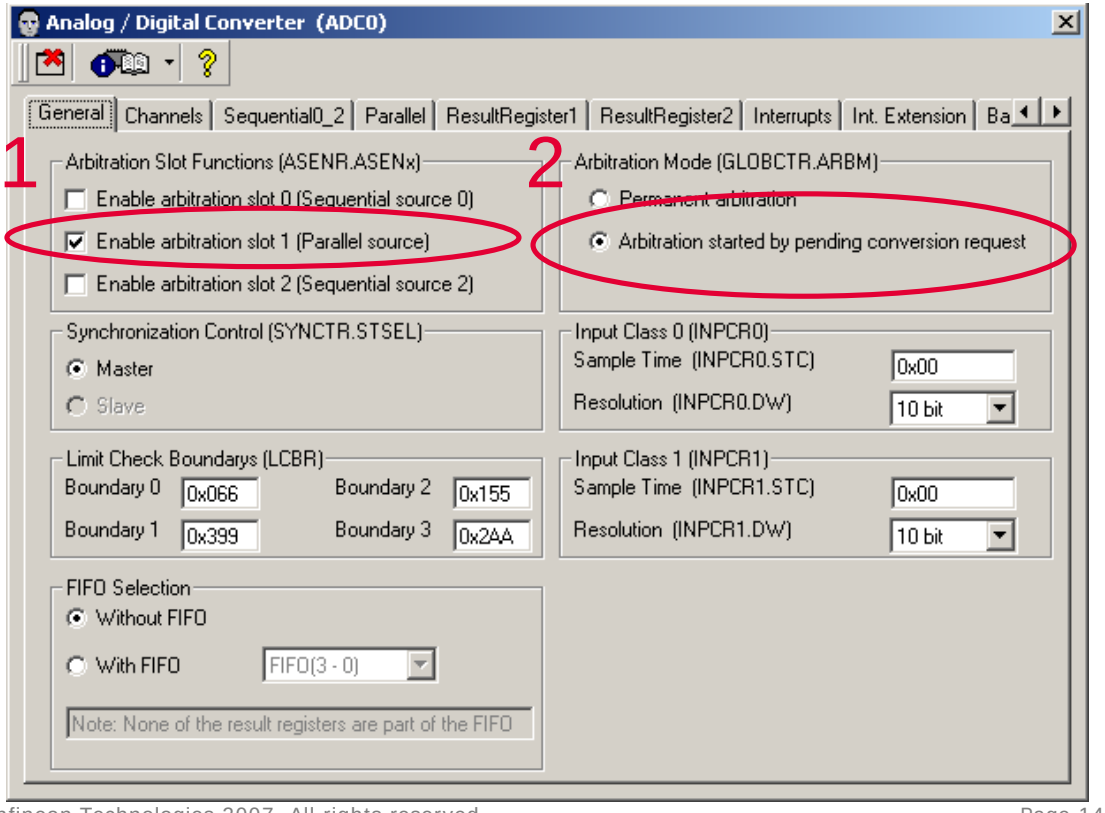
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Click on ADC0 



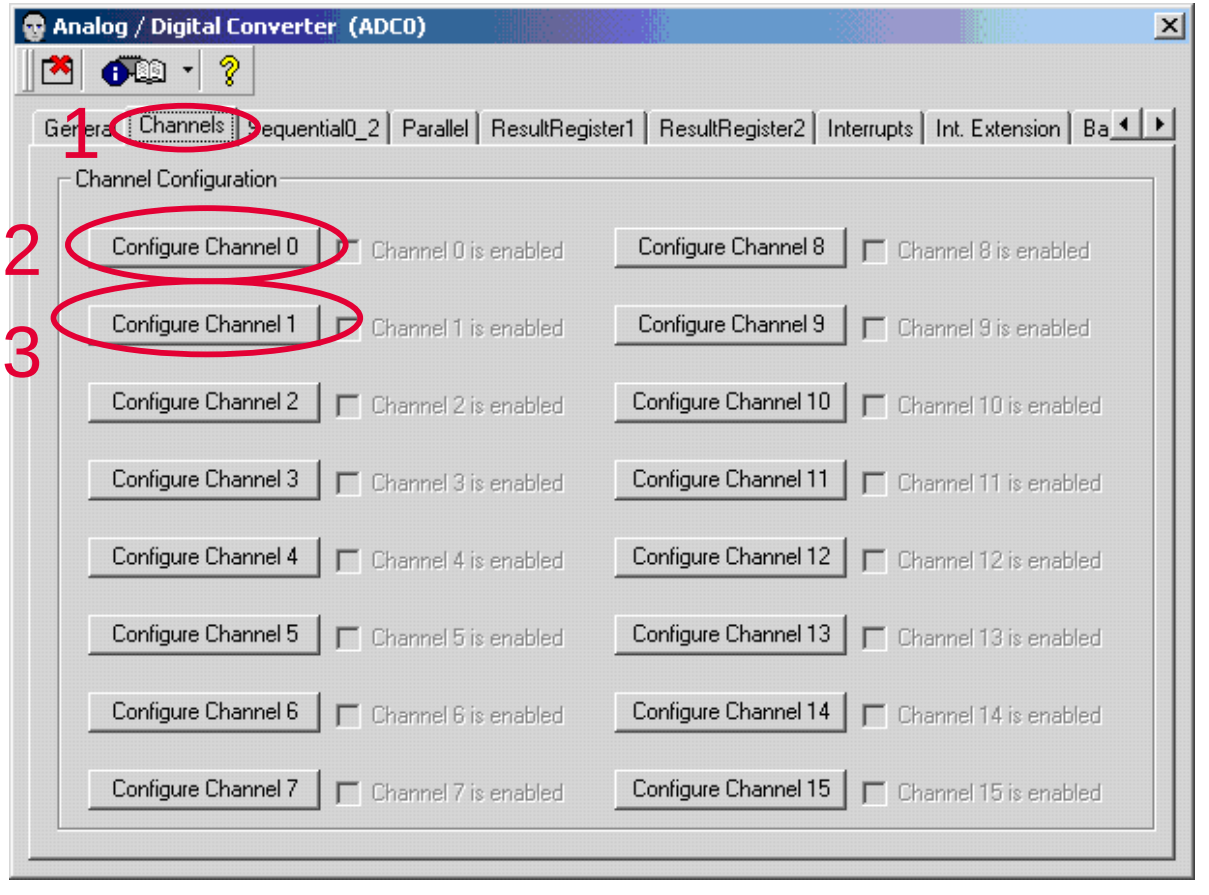
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – General
  - Arbitration Slot Functions – enable ‘arbitration slot 1’ only
  - Arbitration Mode – enable ‘Arbitration started by pending conversion request’
  - Others- default



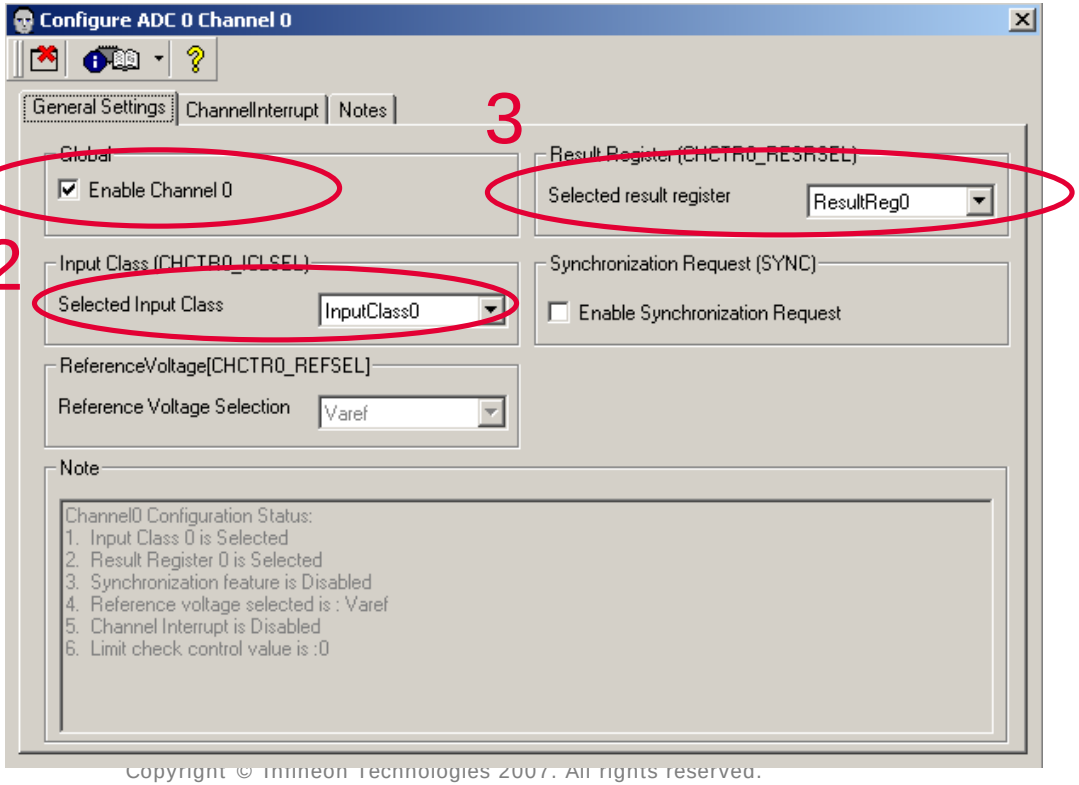
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Channels
  - Click on ‘Configure Channel 0’ and ‘Configure Channel 1’



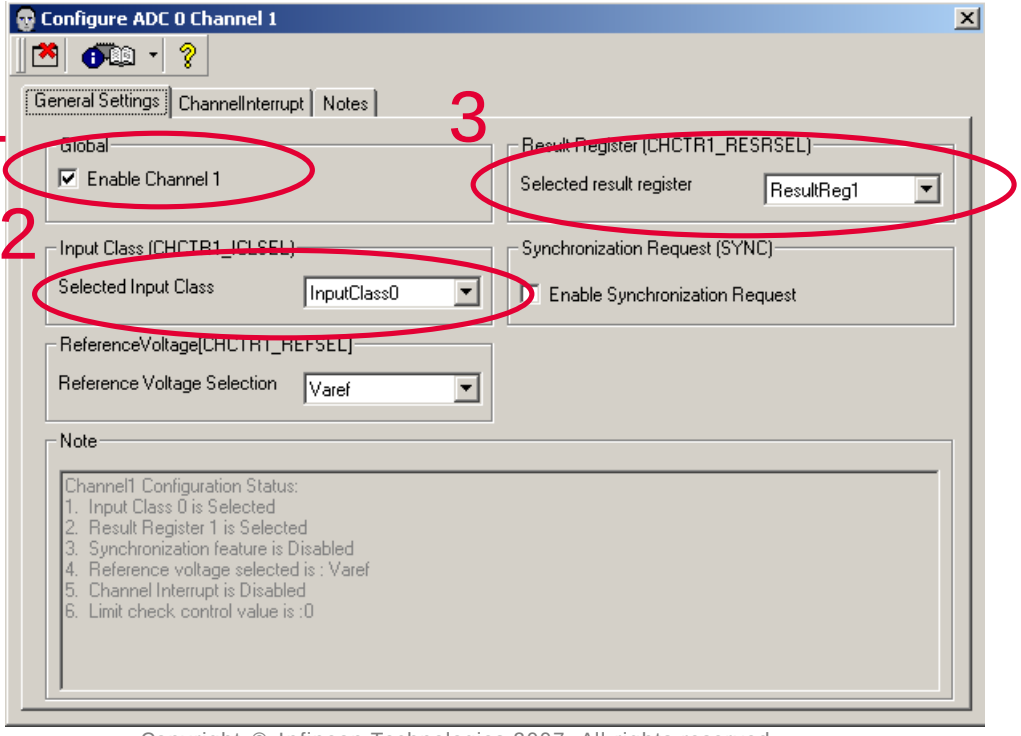
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Channels
  - Enable Channel 0
  - Input Class – select ‘InputCalsss0’
  - Result Register - select ‘ResultReg0’



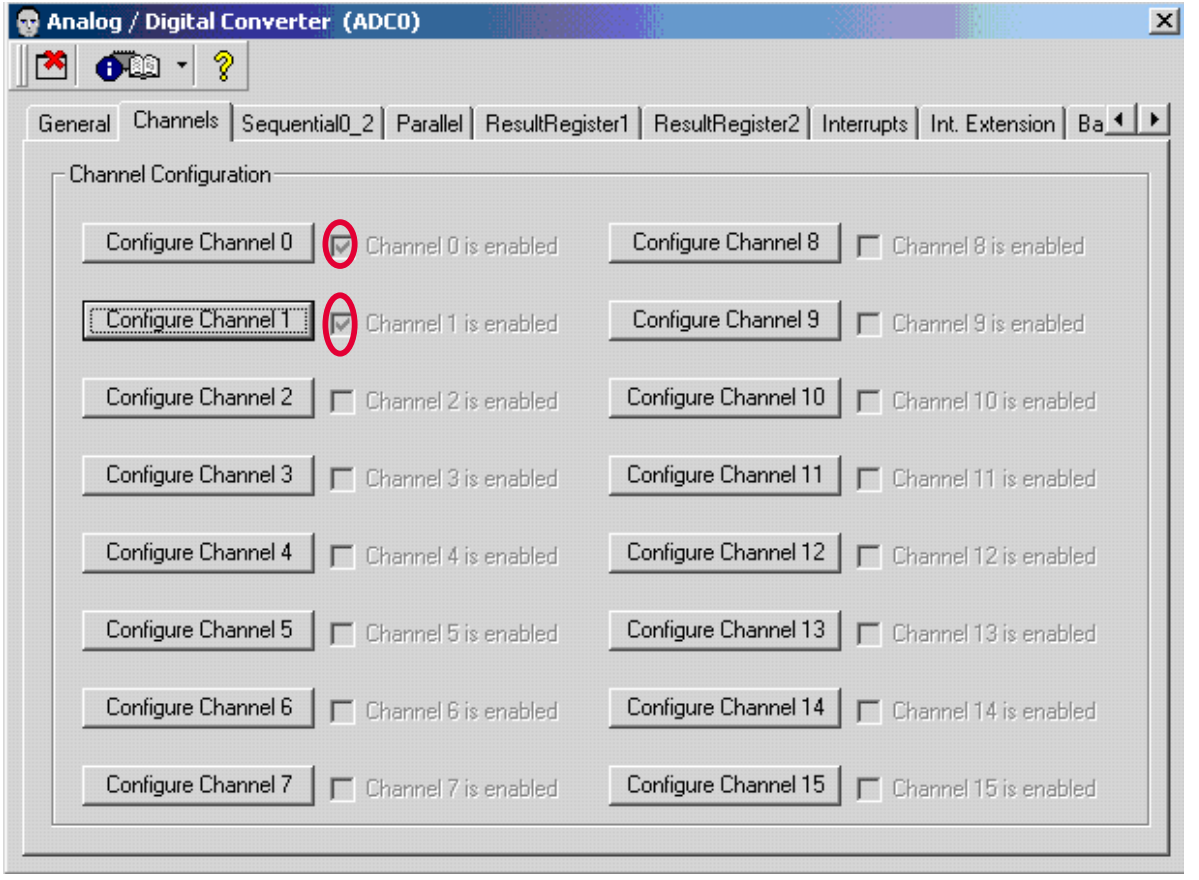
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Channels
  - Enable Channel 1
  - Input Class – select ‘InputClass0’
  - Result Register – select ‘ ResultReg1’



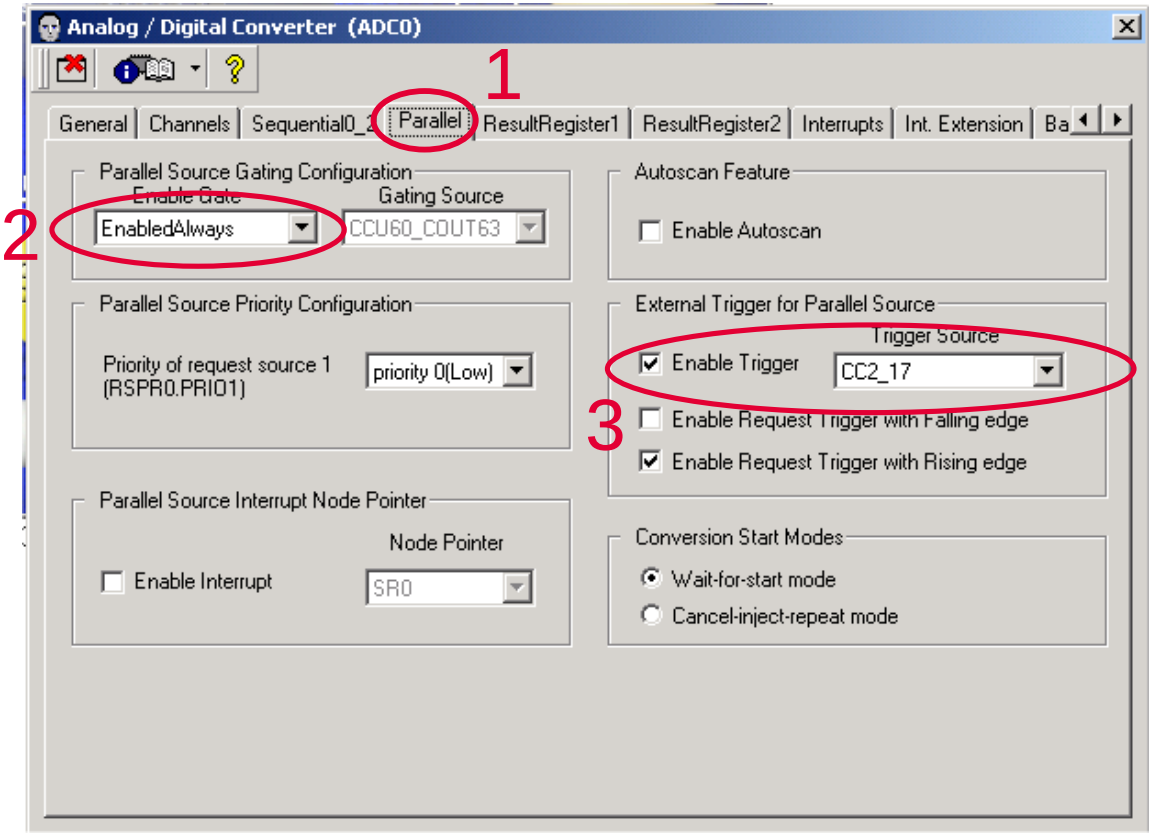
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Channels
  - Now Channel 0 and Channel 1 are selected



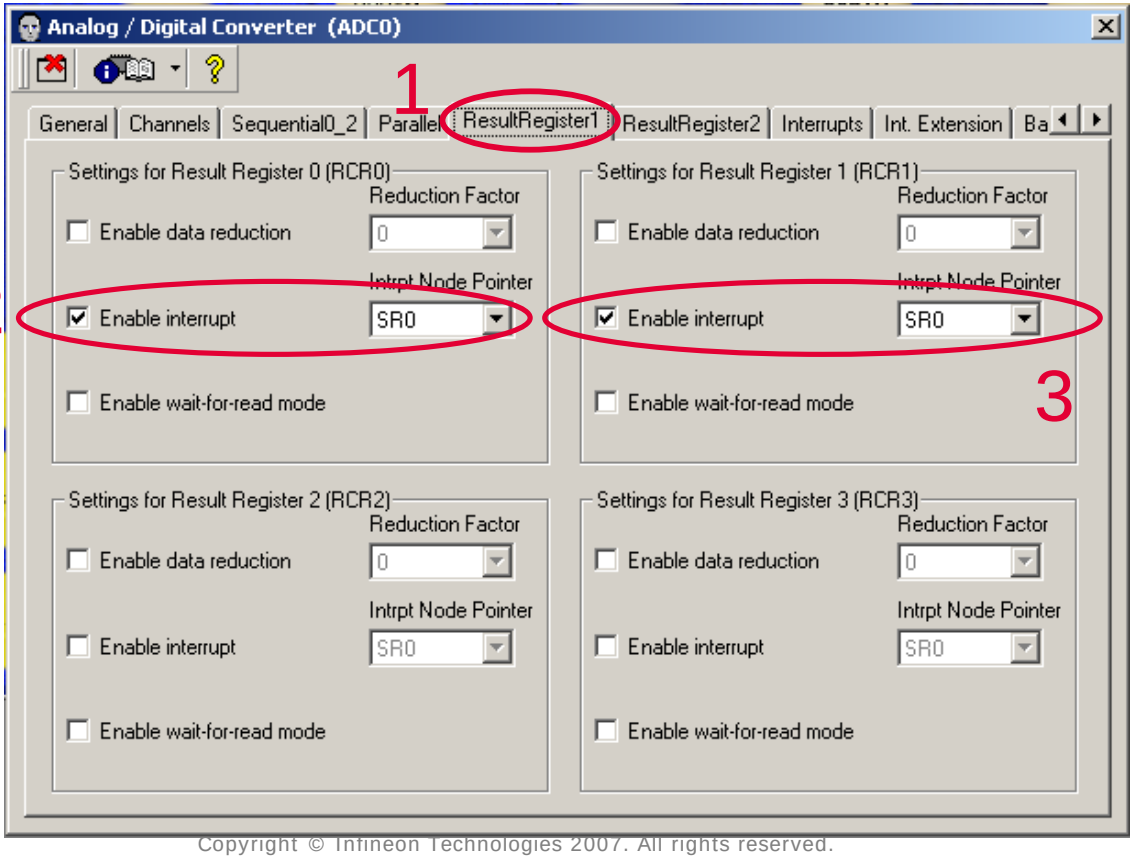
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Parallel
  - Parallel Source Gating Configuration – select ‘EnabledAlways’
  - External Trigger for Parallel Source – select ‘CC2_17’



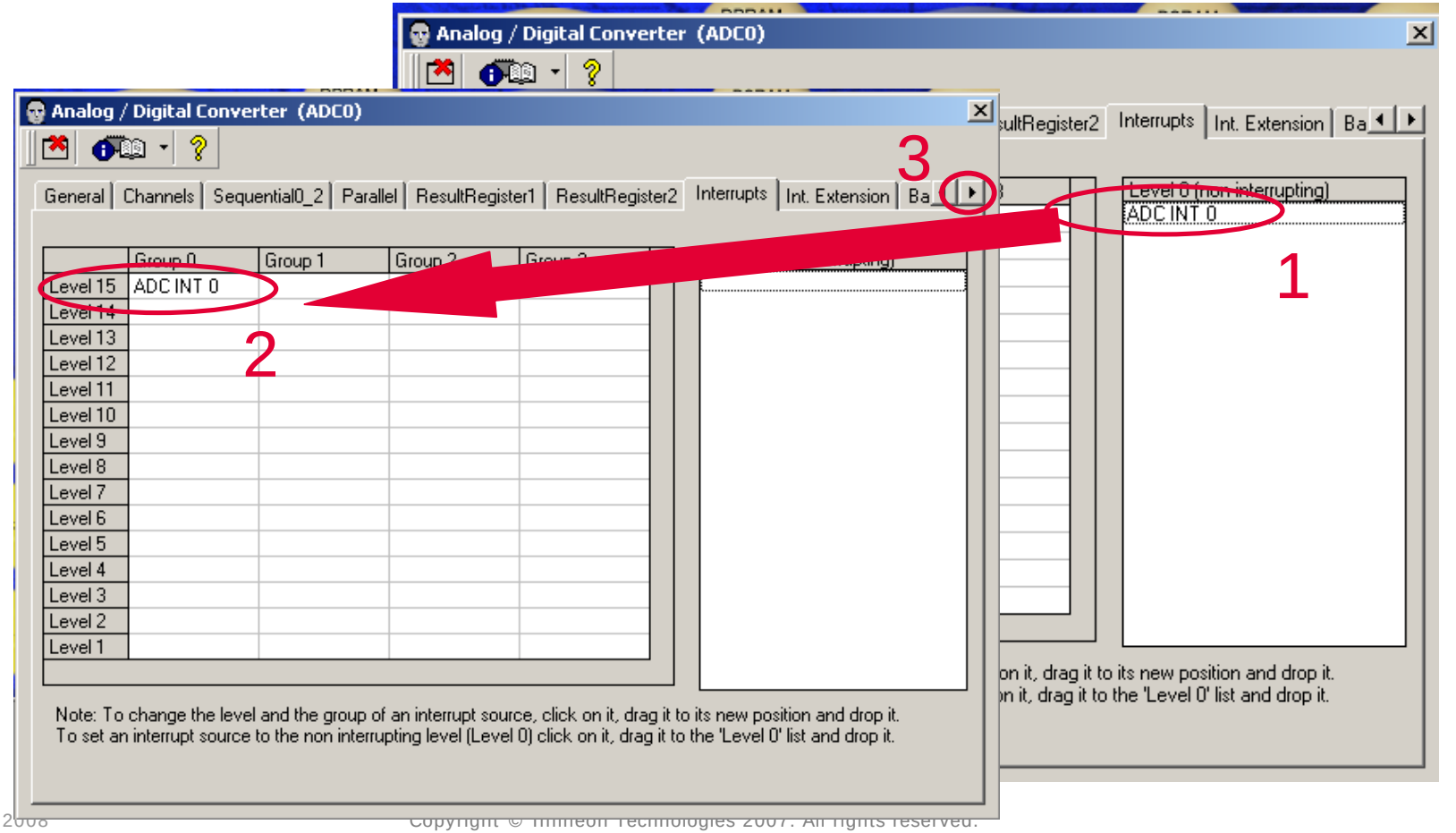
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Result Register1
  - Settings for Register 0 – enable interrupt SR0
  - Settings for Register 1 – enable interrupt SR0



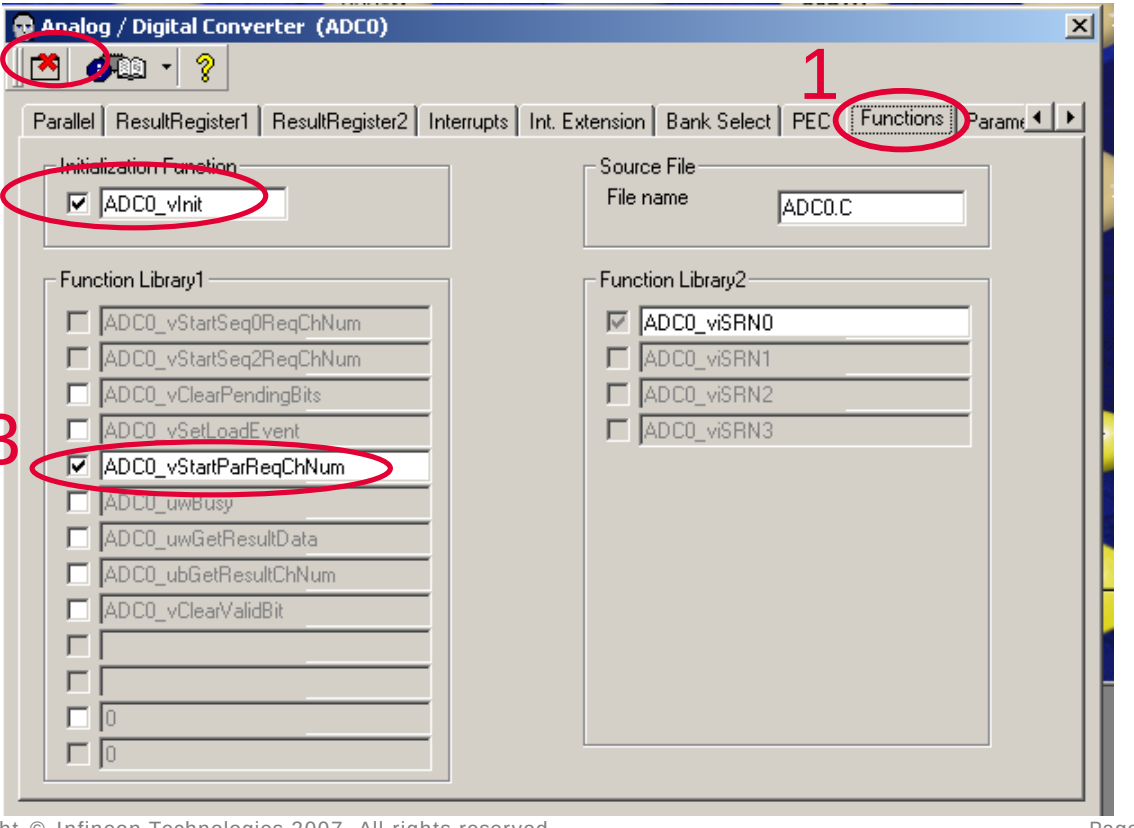
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Interrupts
  - Drag ‘ADC INT 0’ from Level 0 to Level 15, Group 0



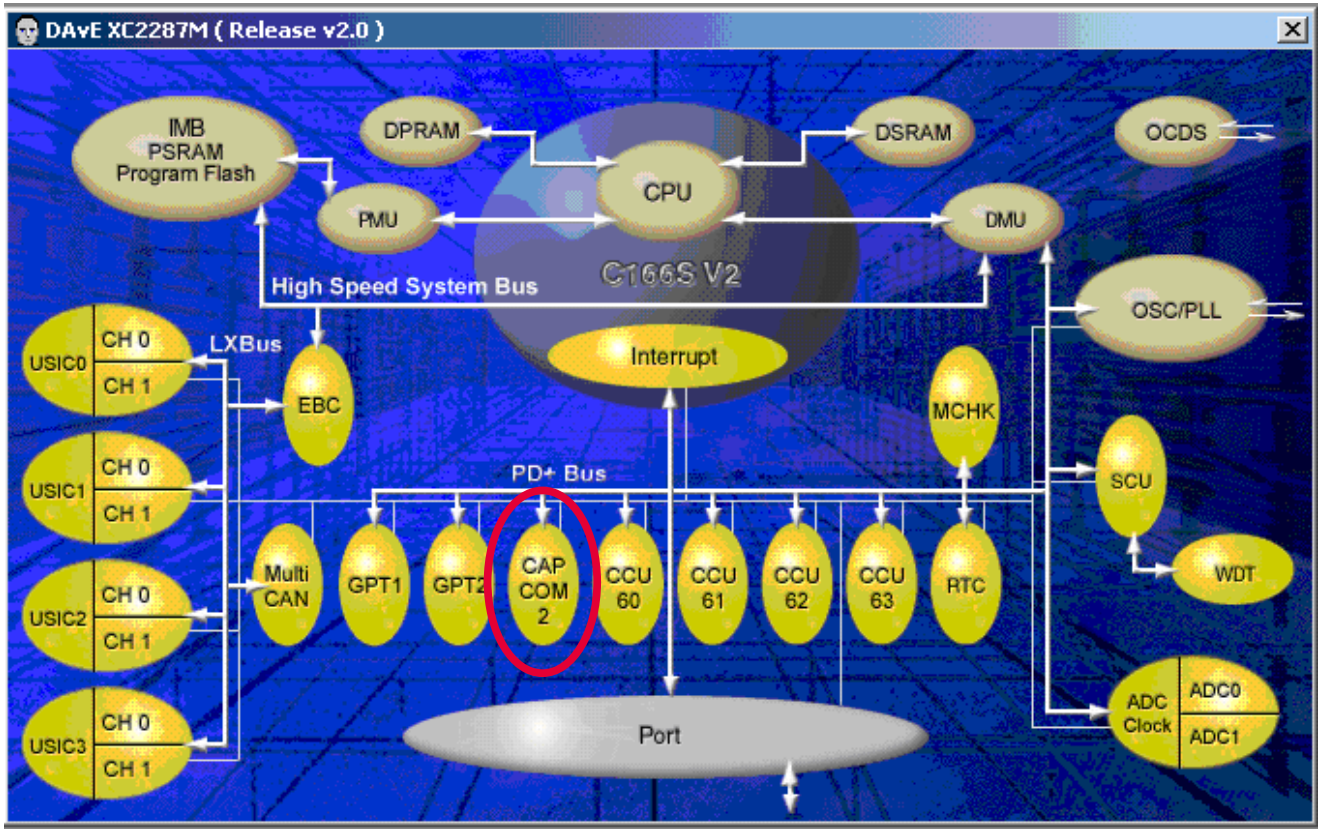
HOT Exercise ADC_CC2 - DAvE Configurations ADC Settings (cont.)

- Configure ADC0 – Functions
  - Click on ‘ADC_vInit’
  - Click on ‘ADC0_vStartParReqChNum’
  - Click on 



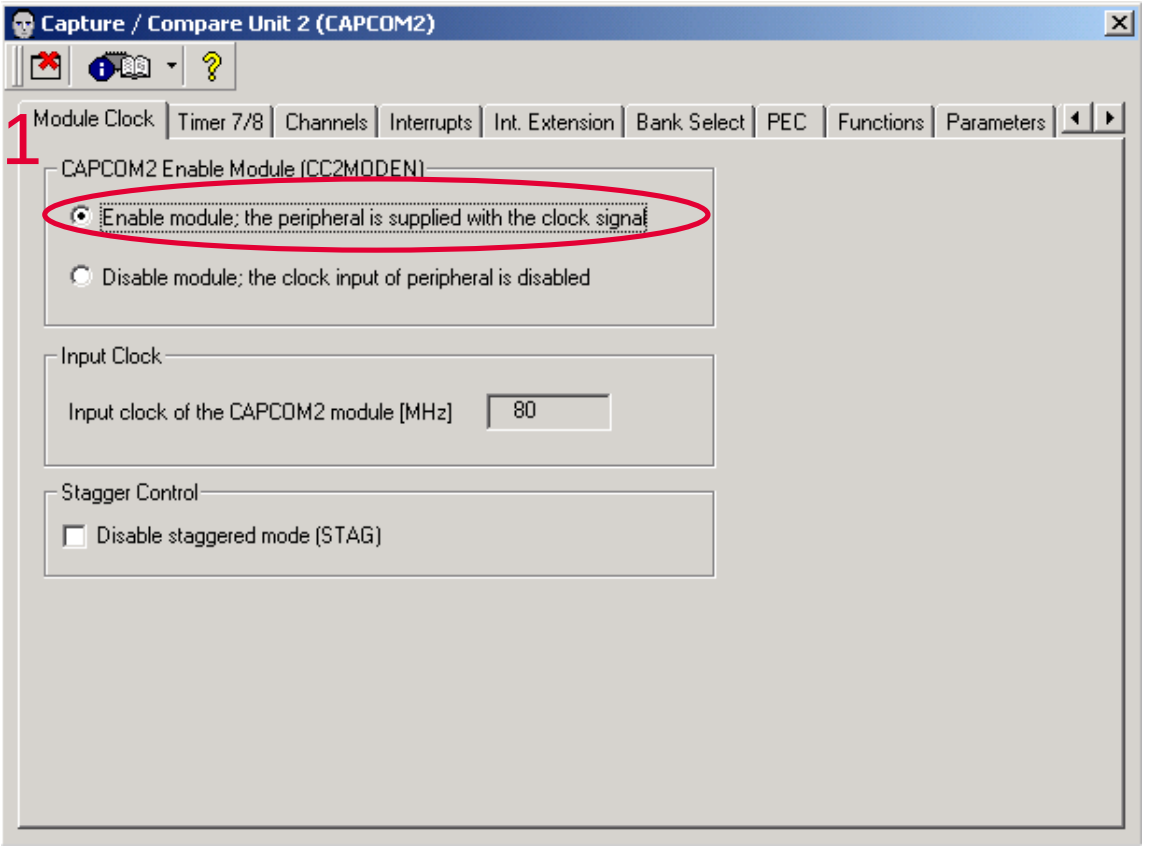
HOT Exercise ADC_CC2 - DAvE Configurations CC2 Settings

- Click on ‘CAPCOM2’ 



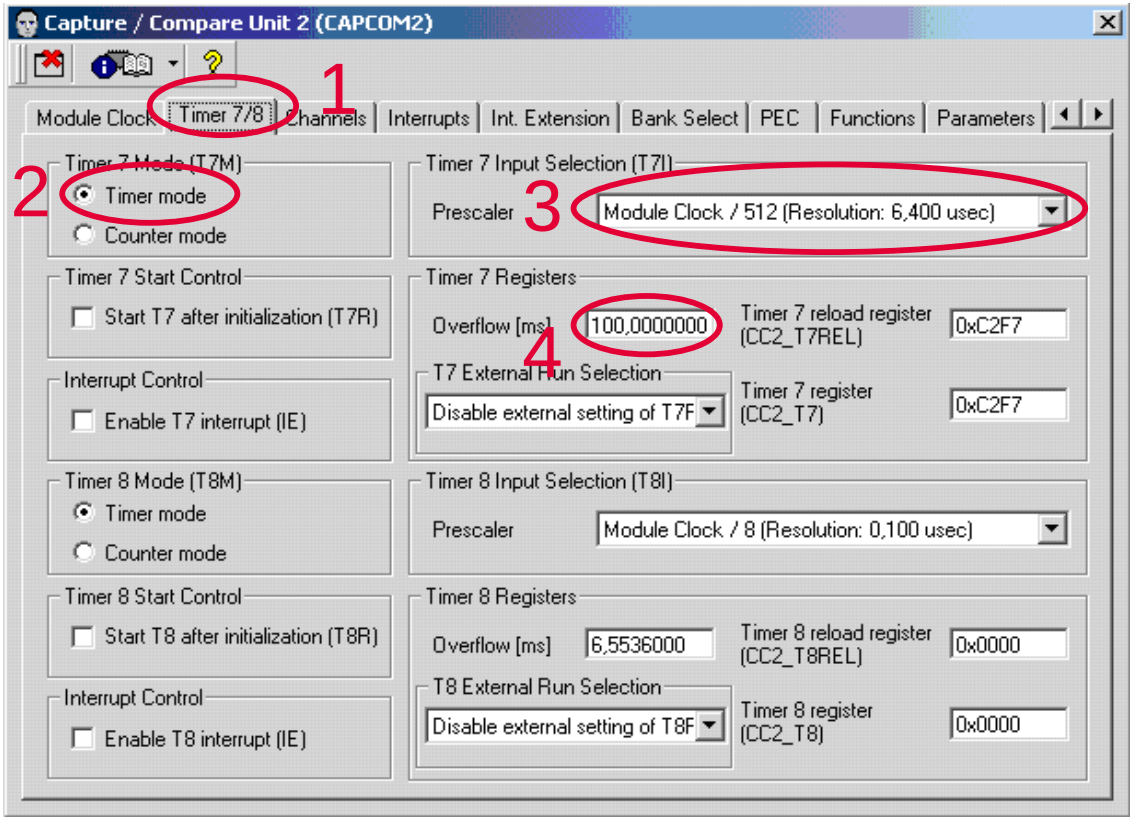
HOT Exercise ADC_CC2 - DAvE Configurations CCU60 Settings (cont.)

- Configure CC2 – Module Clock
  - Enable module



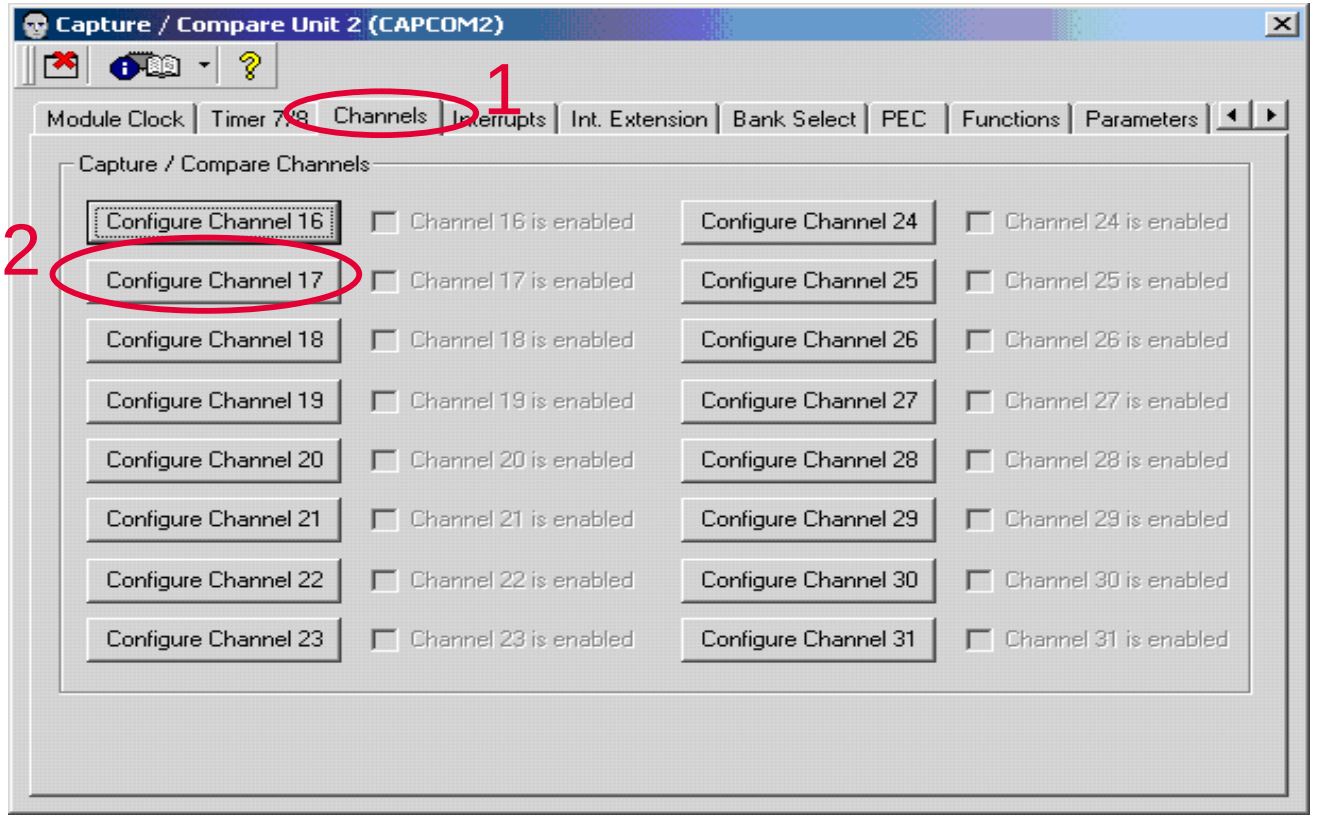
HOT Exercise ADC_CC2 - DAvE Configurations CC2 Settings (cont.)

- Configure CC2 – Timer 7
  - Timer 7 Mode (T7M) – select ‘Timer mode’
  - Timer 7 Register – set as ‘100,00ms’



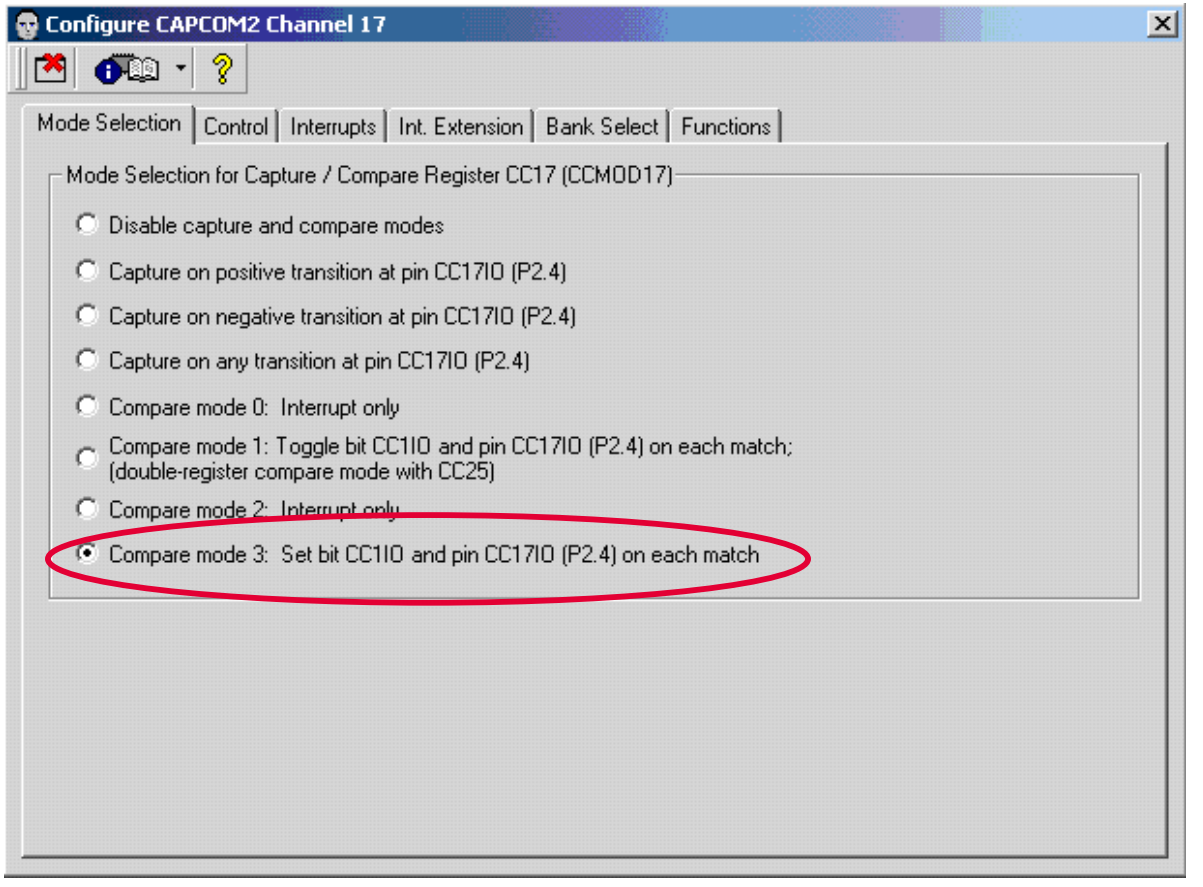
HOT Exercise ADC_CC2 - DAvE Configurations CC2 Settings (cont.)

- Configure CC2 – Channels
  - Channels – click on ‘Channels’
  - Configure Channel 17 – click on ‘Configure Channel 17’



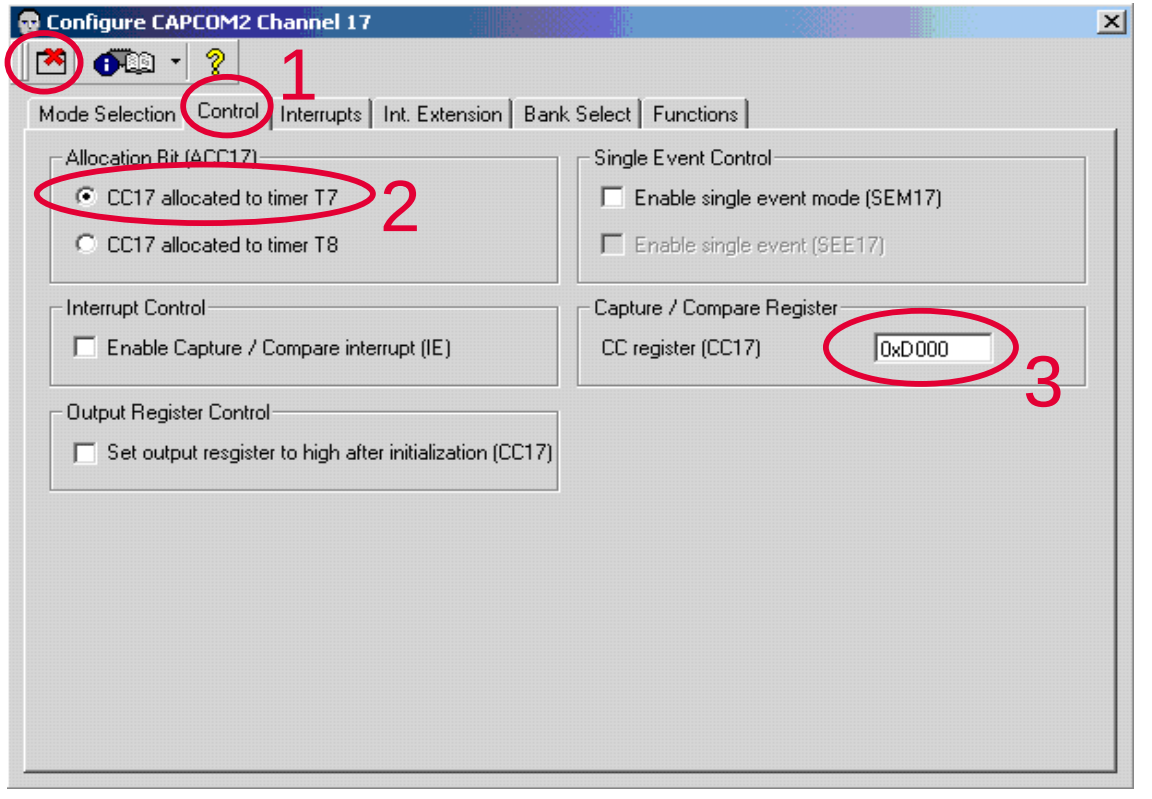
HOT Exercise ADC_CC2 – DAvE Configurations CC2 Settings (cont.)

- Mode selection - click on ‘Compare mode 3 ’



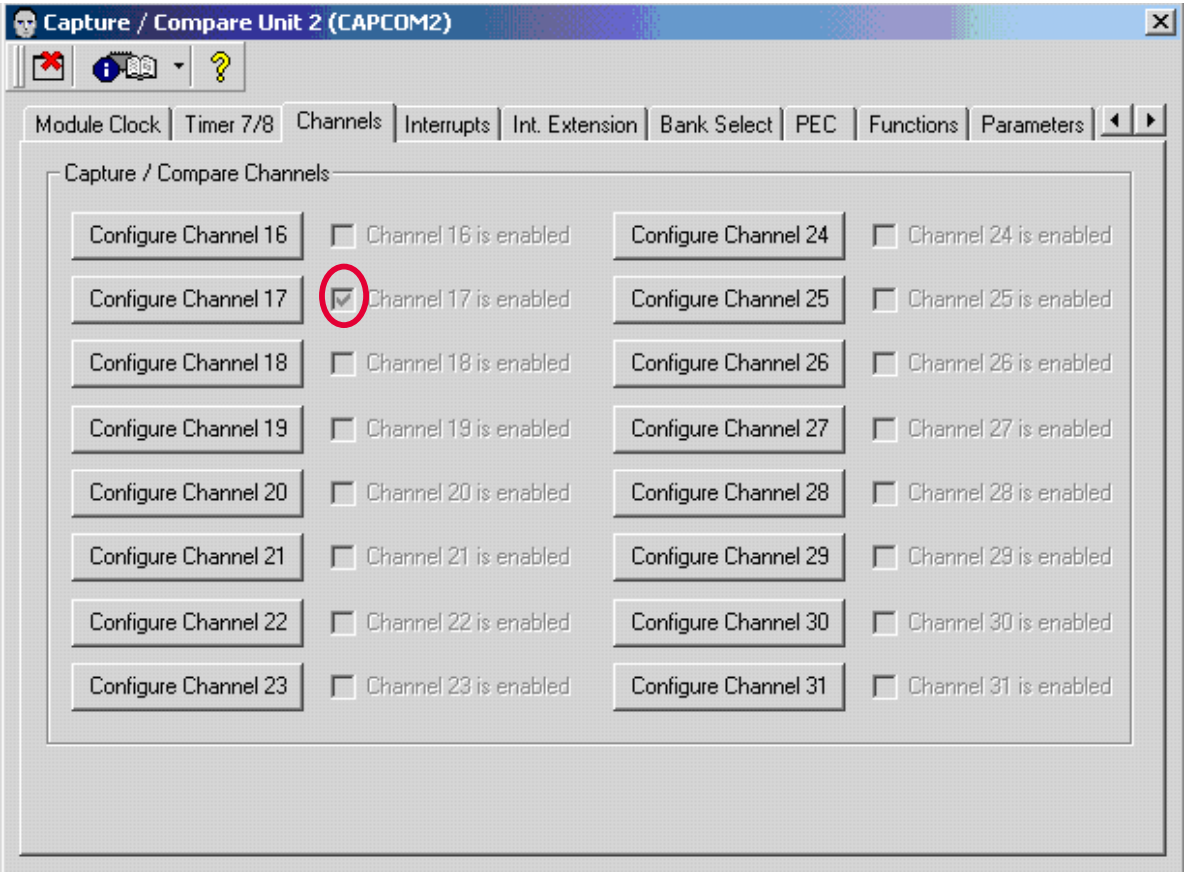
HOT Exercise ADC_CC2 - DAvE Configurations CC2 Settings (cont.)

- Configure CC2 – Channels
  - Configure CC17 – write ‘0xE000 ’
  - Click on 



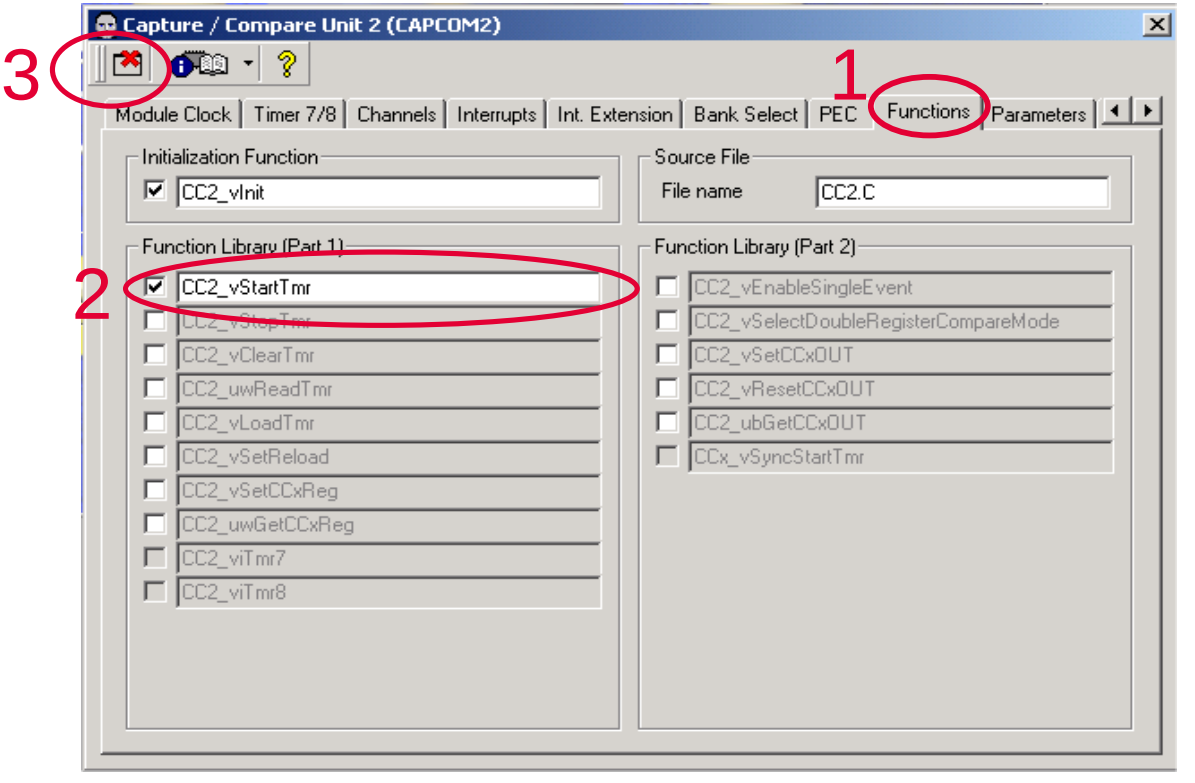
HOT Exercise ADC_CC2 - DAvE Configurations CC2 Settings (cont.)

- Configure CC2 – Channels
  - Now Channel 17 is selected



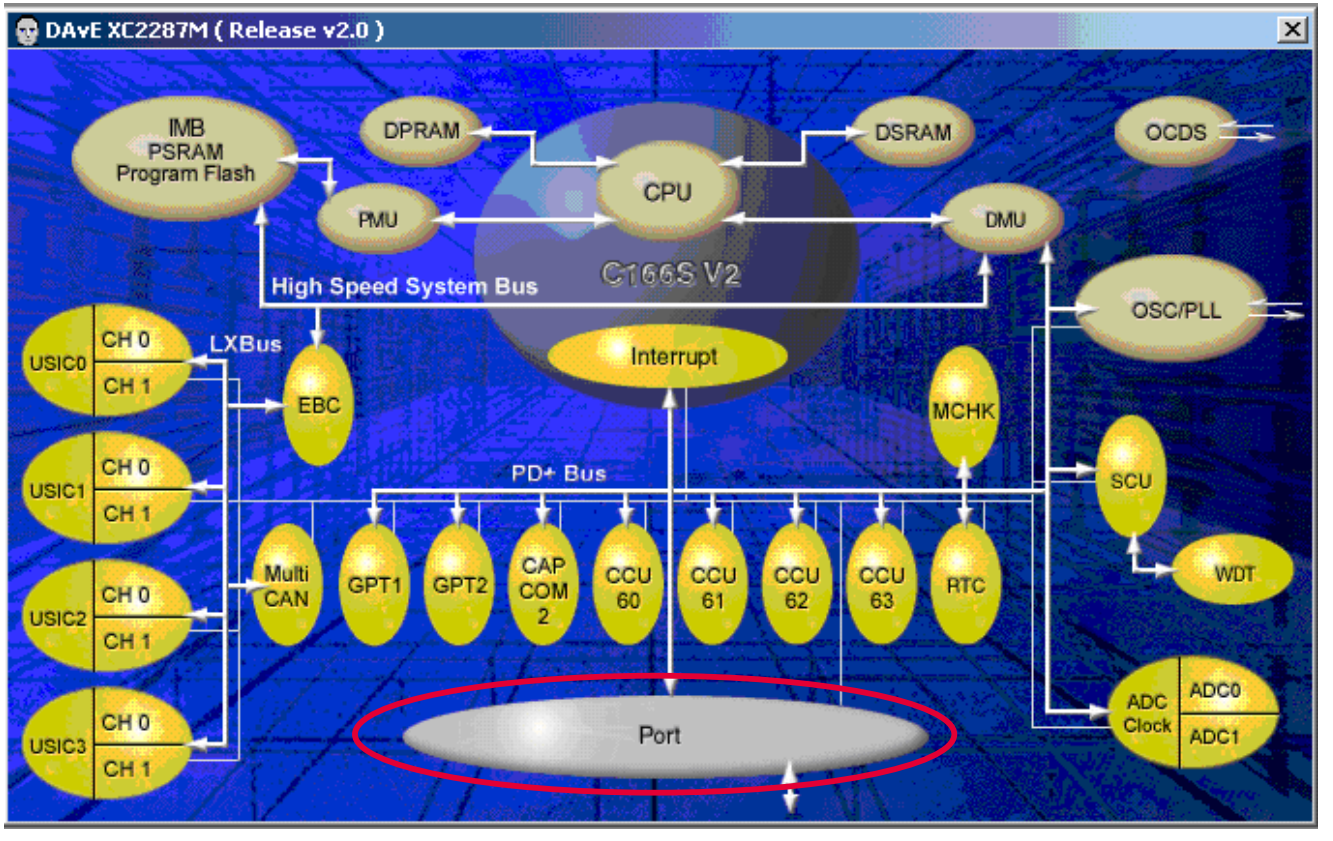
HOT Exercise ADC_CC2 - DAvE Configurations CC2 Settings (cont.)

- Configure CC2 – Functions
  - Function Library (Part1) – configure ‘CC2_vStartTmr’
  - Click on 



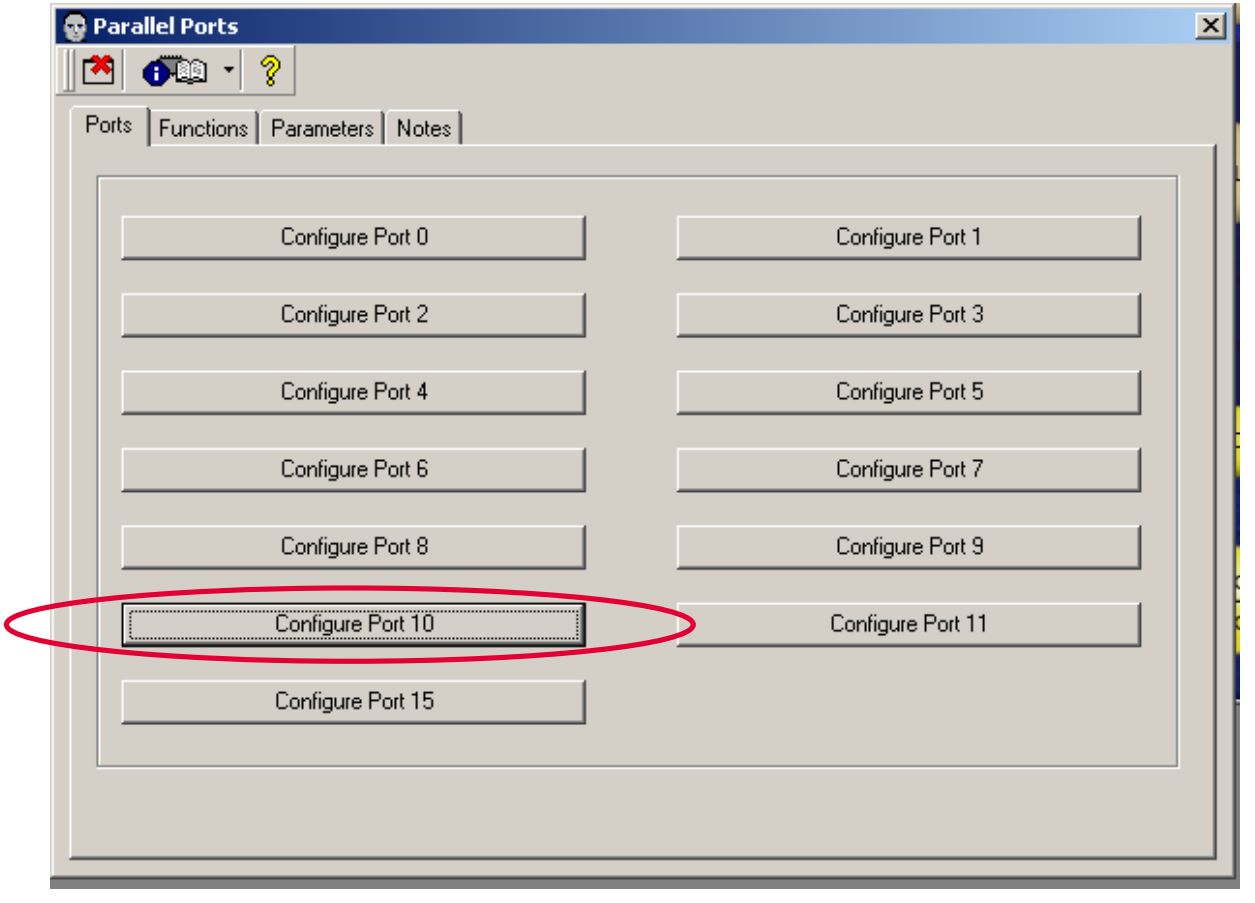
HOT Exercise ADC_CC2 - DAvE Configurations Port Settings

- Click on ‘Port’



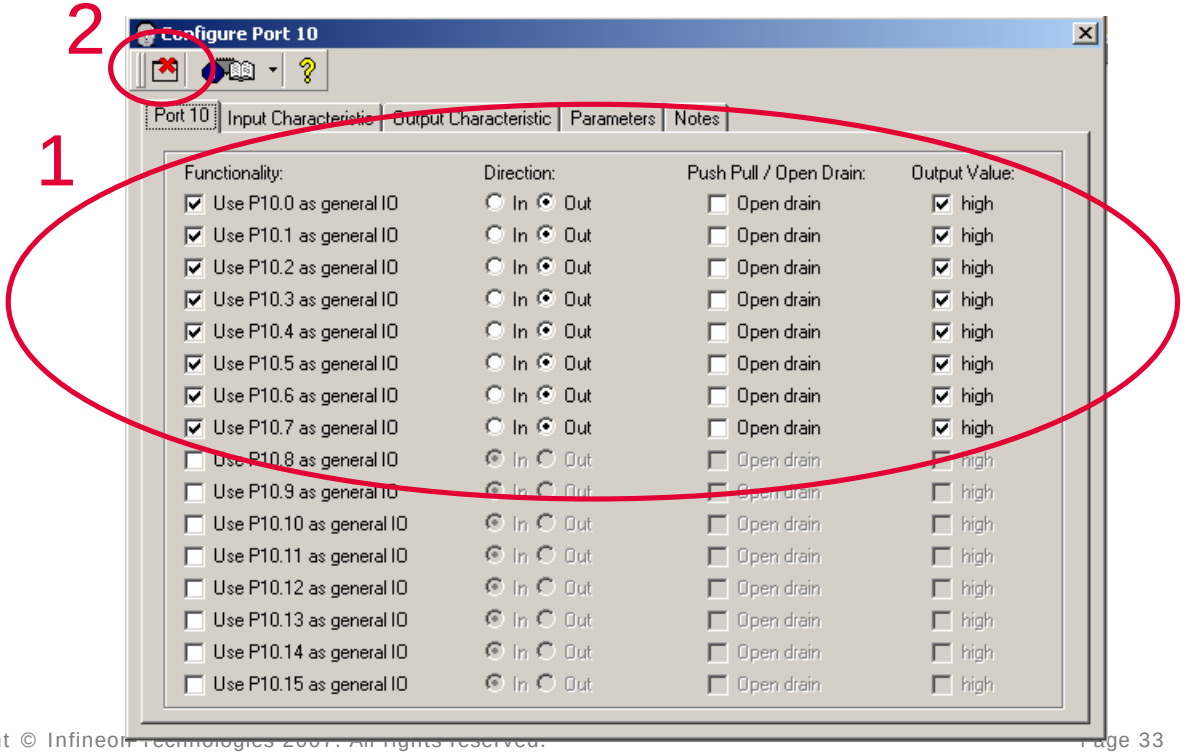
HOT Exercise ADC_CC2 - DAvE Configurations Port Settings (cont.)

- Configure – Ports
  - Click on ‘Configure Port 10’



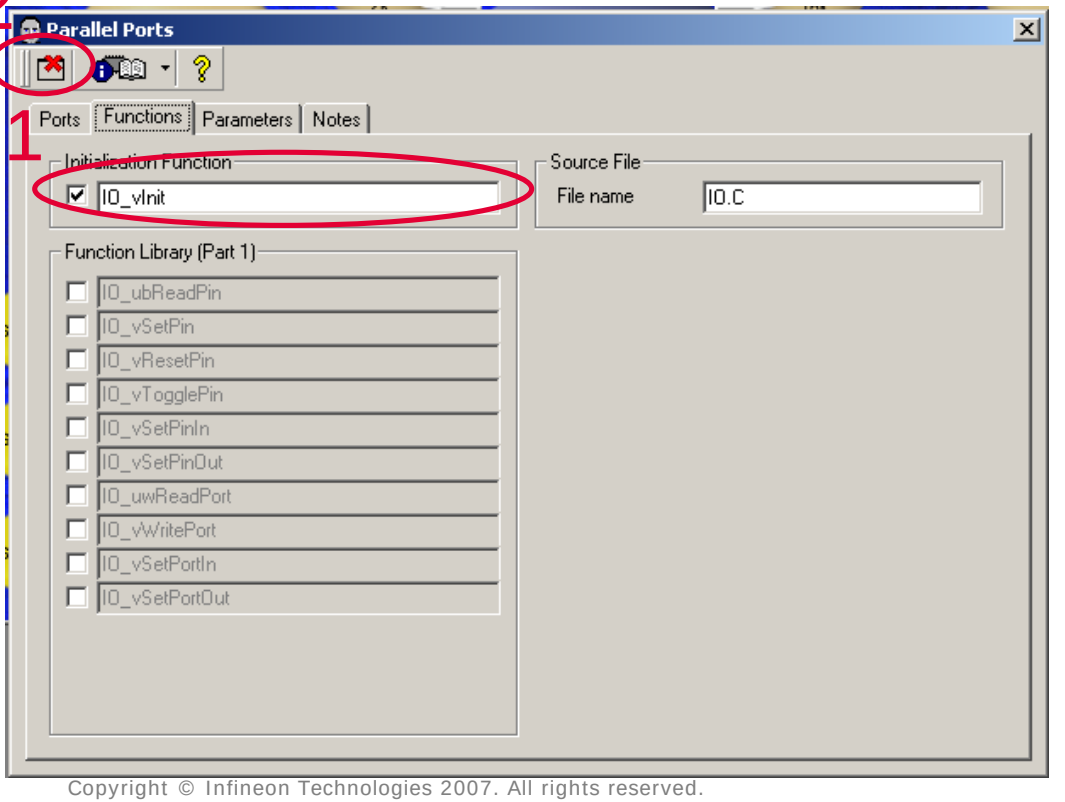
HOT Exercise ADC_CC2 - DAvE Configurations Port Settings (cont.)

- Configure Ports – Port 10
  - Functionality - click on ‘P10.0(to P10.7) as general IO’
  - Direction – click on ‘Out’
  - Output Value – click ‘high’
  - Click on 



HOT Exercise ADC_CC2 - DAvE Configurations Port Settings (cont.)

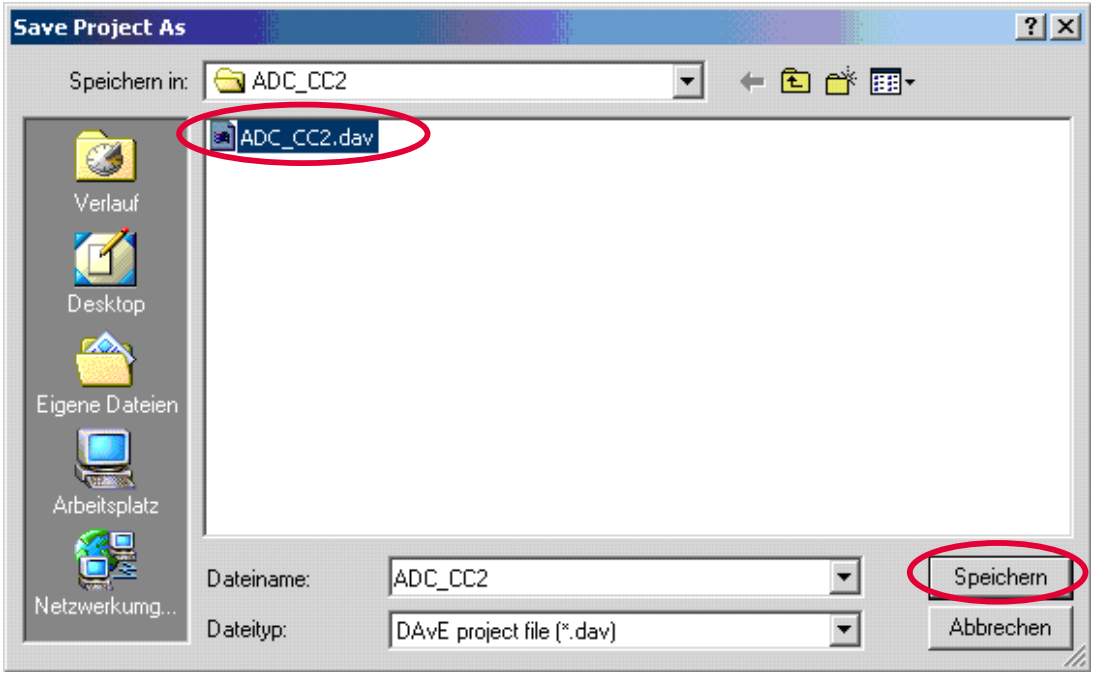
- Configure Ports – Functions
  - Initialization Function - click on ‘IO_vInit’
  - Click on 



HOT Exercise ADC_CC2 - DAvE Configurations
Save DAvE Project

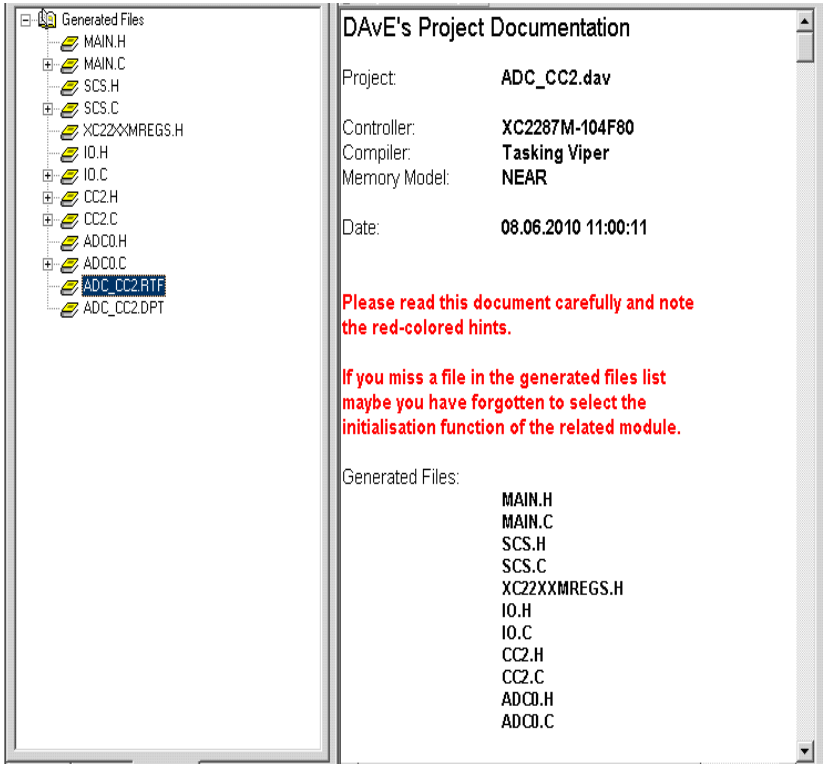
- Save your DAvE Project File
  - Go to **File → Save (or Save As)** or click on 
  - Filename:

    “c:\IFX_HOT\XC2287M\Example\ADC_CC2\ADC_CC2.dav”

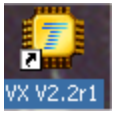


HOT Exercise ADC1 - DAvE Configurations Code Generation

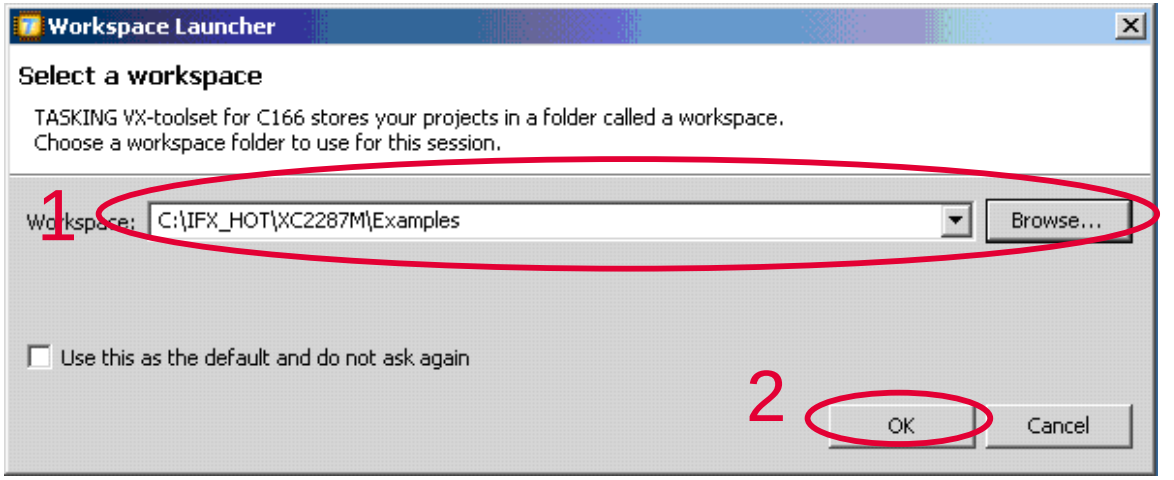
- Let DAvE Generate Code for You
  - Go to **File → generate Code** or click on 
  - DAvE generated code files are
    - MAIN.C, MAIN.H
    - ADC0.C, ADC0.H
    - CC2.C, CCU2.H
    - IO.C, IO.H
    - SCS.C, SCS.H
    - XC22XXREGS.H



HOT Exercise ADC_CC2 – Tasking VX Toolset

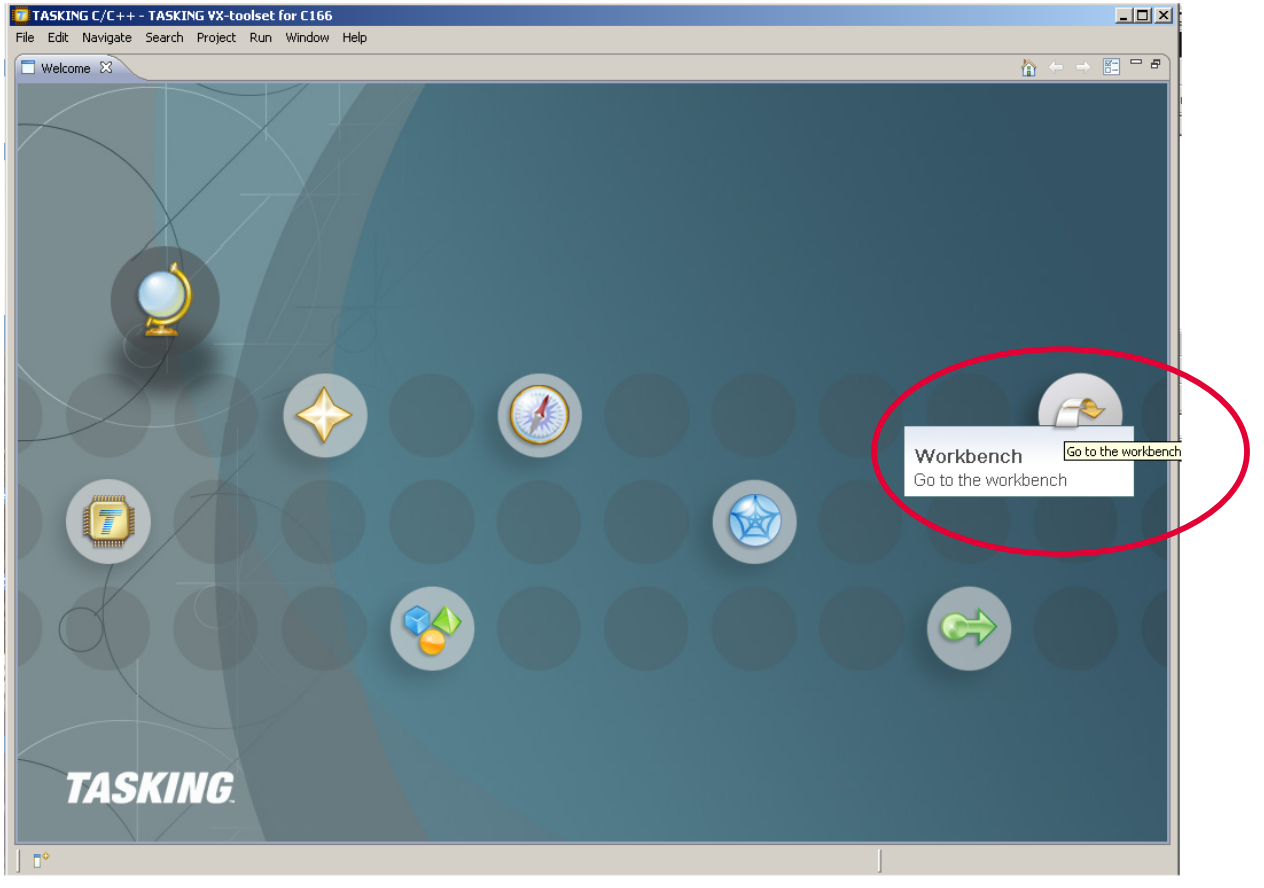
- Create New Work Space
  - Click on 

  - Filename: **“c:\IFX_HOT\XC2287M\Examples”**
  - Click ‘OK’



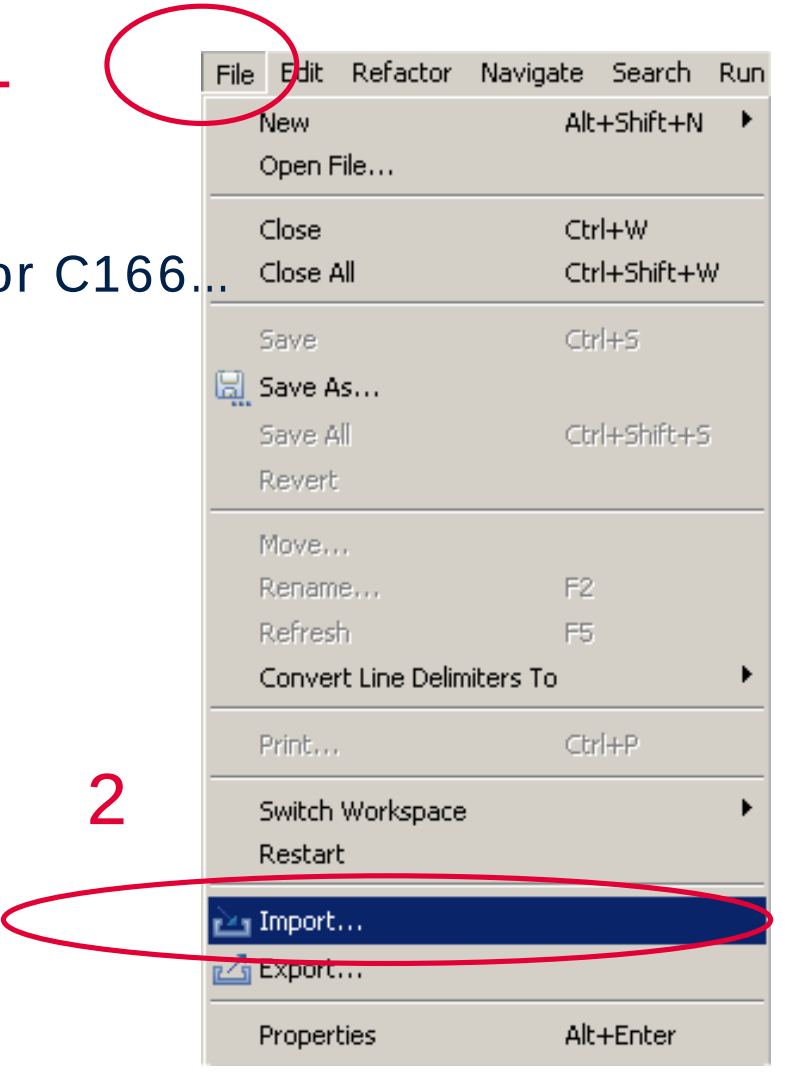
HOT Exercise ADC_CC2 – Tasking VX Toolset

- Create New Project
  - Click on Workbench



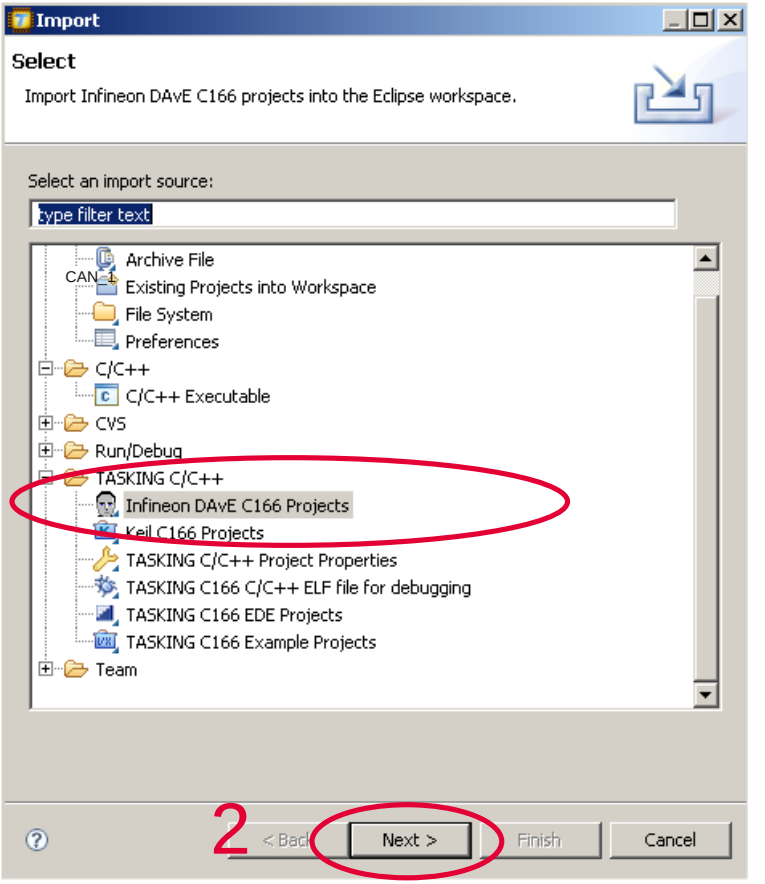
HOT Exercise ADC_CC2 – Tasking VX Toolset

- Import DAVE Project
  - Click on File -> Import
  - Select Tasking VX-toolset for C166...
  - Click ‘OK’



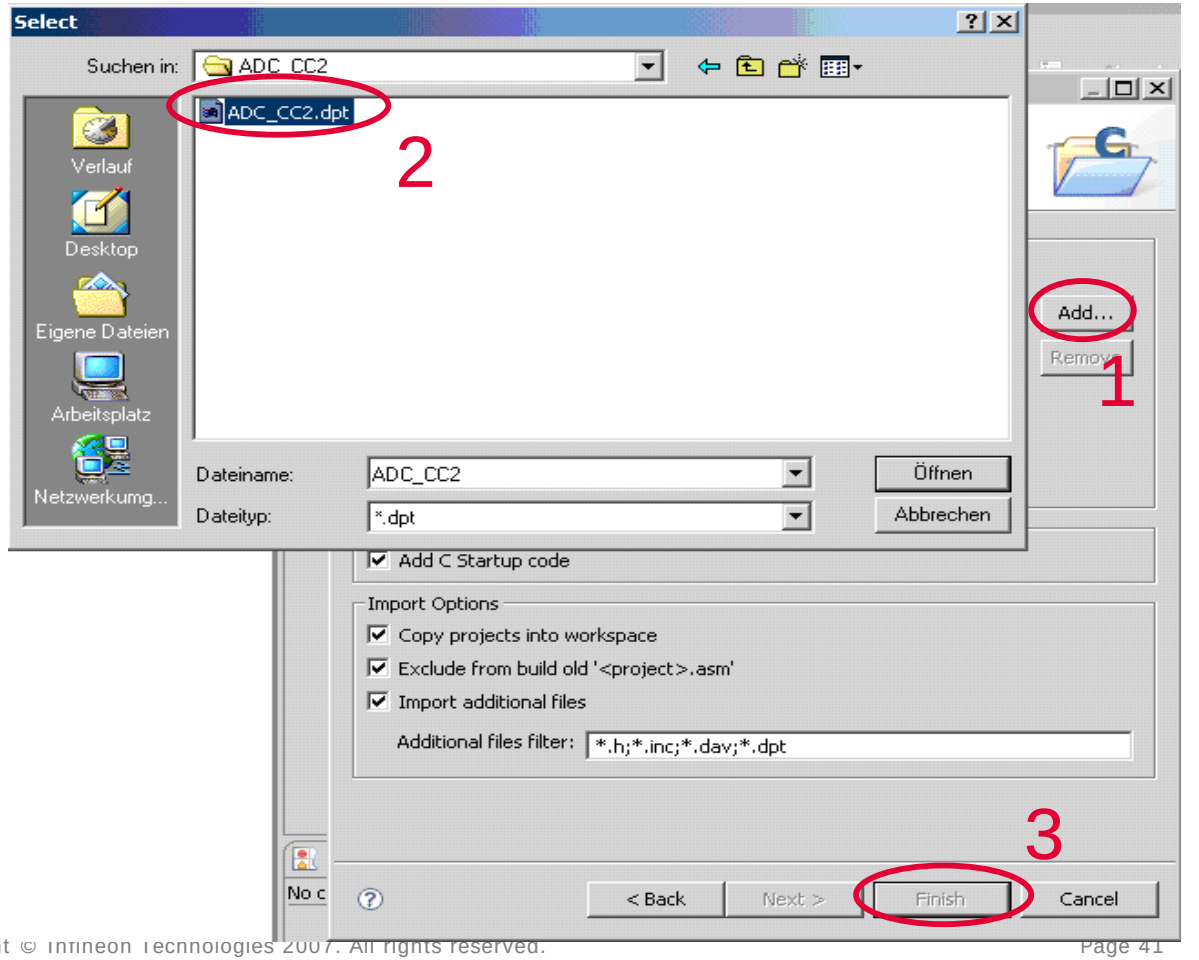
HOT Exercise ADC_CC2 – Tasking VX Toolset

- Import DAVE Project
  - Click `Infineon DAvE C166 Project´
  - Click ‘Next’



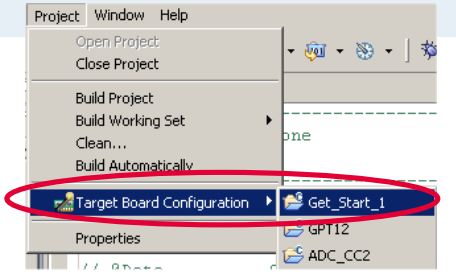
HOT Exercise ADC_CC2 – Tasking VX Toolset

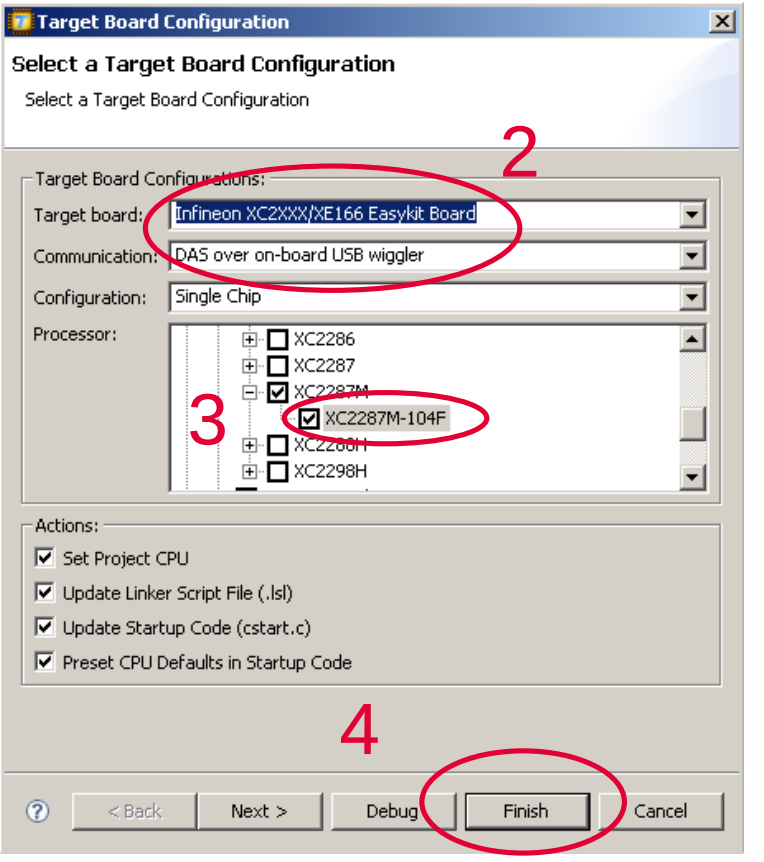
- Import DAvE Project
  - Add Dave Project ‘ADC_CC2´
  - Click `Finish´



HOT Exercise ADC_CC2 – Tasking VX Toolset

- Configure Target Board 
  - Select the project in the navigator
  - Select ‘Project/Target Board Configuration’
  - Select ‘Infineon XC2000/XE166 Easykit Board’
  - Choose `XC2287M-104F´
  - Click `Finish´





HOT Exercise ADC_CC2 – Tasking VX Toolset

- Software Hint
  - DAvE doesn't change code that is inserted in the 'USER CODE' sections if you let DAvE regenerate the code.

    Therefore, whenever adding code to the generated code, write it into a 'USER CODE' section.

    The code you really have to add looks like this:

```
while(1)
{
// USER CODE BEGIN (Main,4)

BlinkLED();

// USER CODE END
}
```

HOT Exercise ADC_CC2 – Tasking VX Toolset
Add User Code – MAIN.C

```
void main(void)
{
  // USER CODE BEGIN (Main,2)
  // USER CODE END

  MAIN_vInit();

  // USER CODE BEGIN (Main,3)

  ADC0_vStartParReqChNum(0x0001);      //Conversion request: convert channel 0
  CC2_vStartTmr_CC2_TIMER_7();     //Start timer 7 of CC2

  // USER CODE END

  while(1)
  {

   // USER CODE BEGIN (Main,4)

   // USER CODE END

  }

} //  End of function main
```

HOT Exercise ADC_CC2 – Tasking VX Toolset
Add User Code – ADC0.C (ISR)

```
_interrupt(ADC0_SRN0INT)  void ADC0_viSRN0(void)
{

   if((ADC0_EVINFR & 0x0100) == 0x0100)    //Result0 event interrupt
   {
      ADC0_EVINCR = 0x0100;     // Clear Result0 event interrupt

    // USER CODE BEGIN (ADC0_viSRN0,20)

          P10_OMRL = 0x00FF;  // LED off
          P10_OMRL = (ADC0_RESRA0 << 4) & 0xFF00; // put ADC value on LED

    // USER CODE END

   }

   if((ADC0_EVINFR & 0x0200) == 0x0200)    //Result1 event interrupt
   {
      ADC0_EVINCR = 0x0200;     // Clear Result1 event interrupt
    // USER CODE BEGIN (ADC0_viSRN0,21)

    // USER CODE END

   }
```

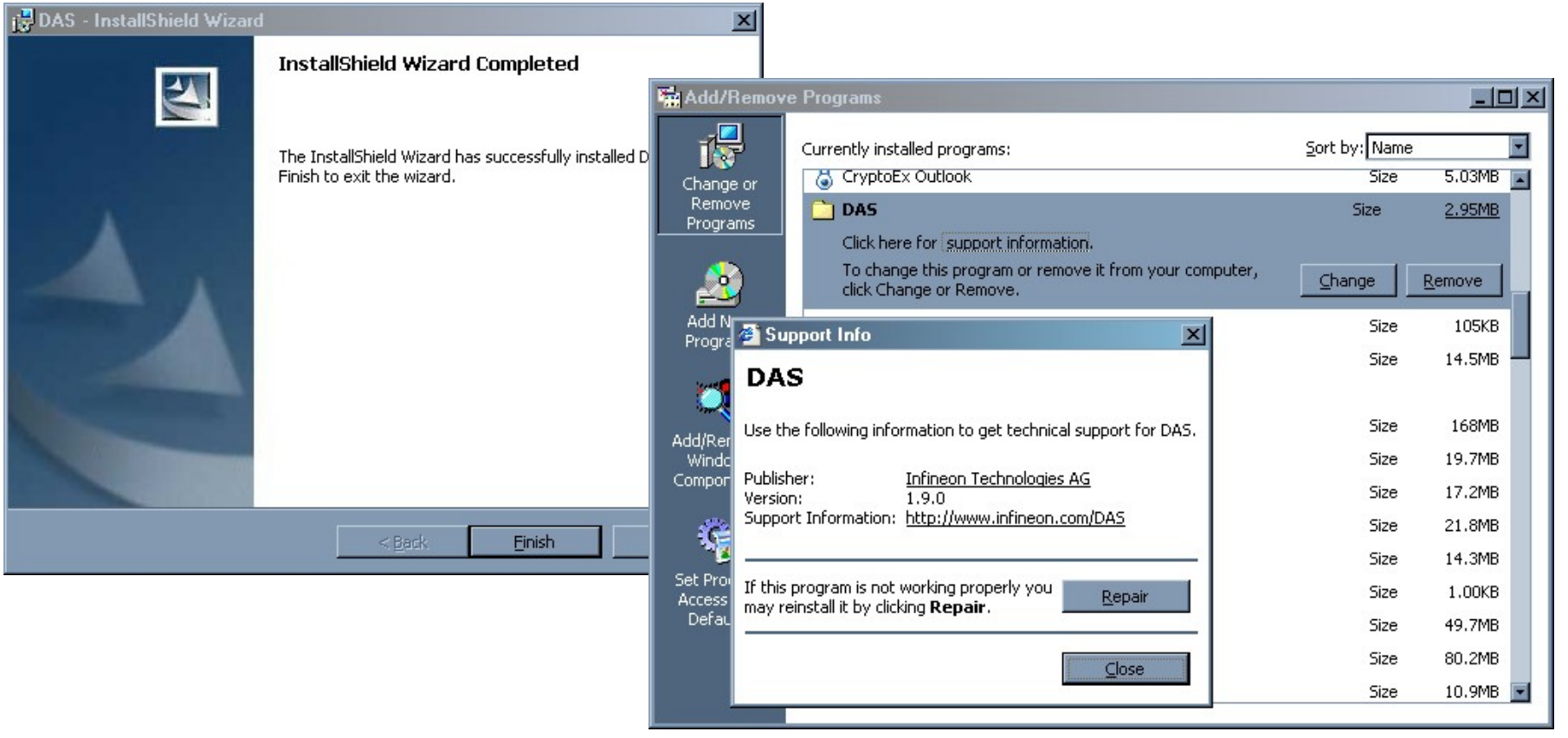
HOT Exercise ADC_CC2 – Tasking VX Toolset Build Project

1

- Click on ‘Build Project ADC_CC2’

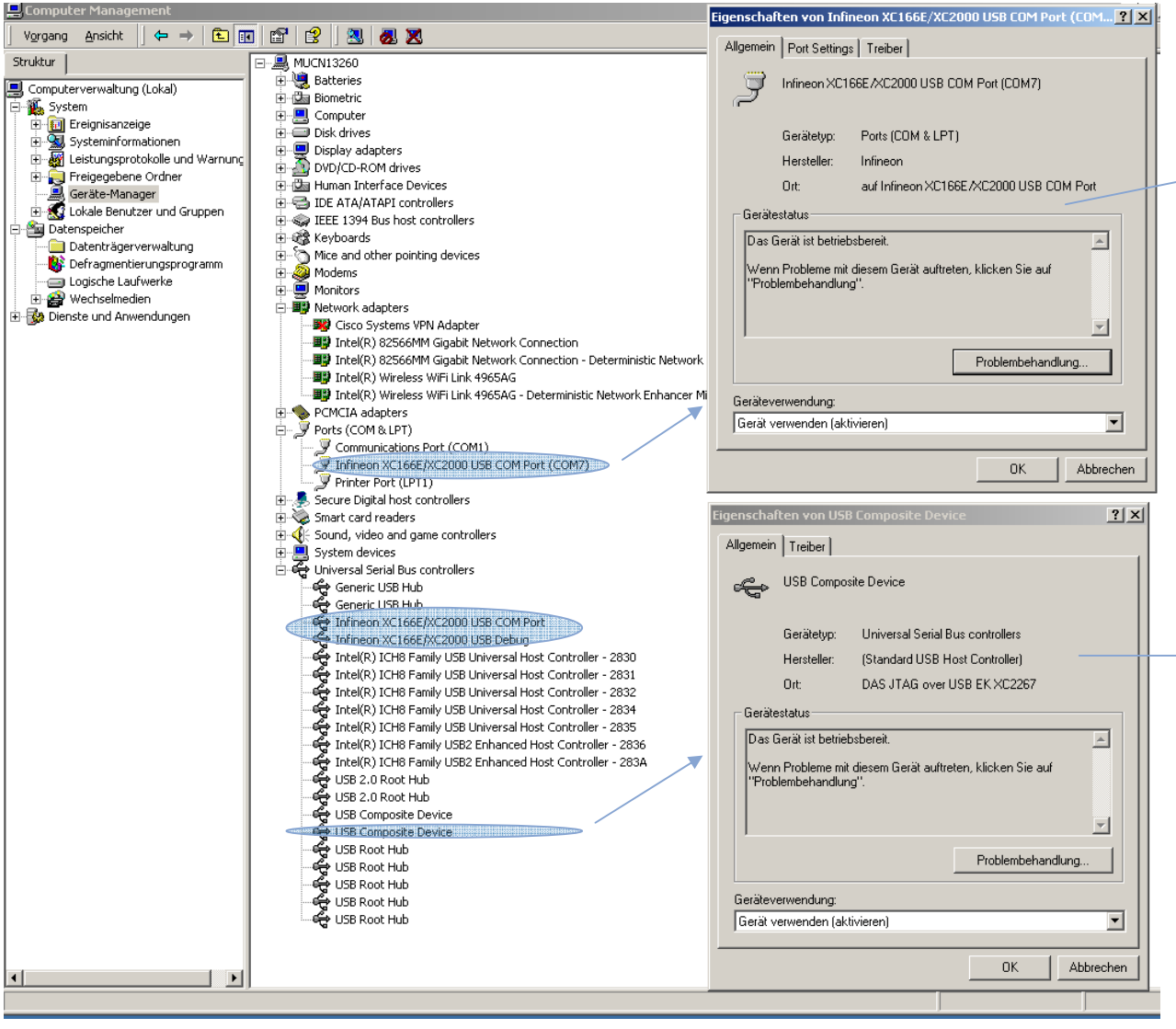
```
C-Build [ADC_CC2]
0 errors, 2 warnings
Compiling SCS.c
c166 W533: ["..\XC22xxMREGS.h"
0 errors, 1 warnings
Linking to ADC_CC2.elf

Time consumed: 3205 ms
**** End of build ****
```

HOT Exercise ADC_CC2 - Device Access Server
Check for the latest DAS version



Note: It is recommended to use the latest DAS version.
Download the latest version at www.infineon.com\DAS

HOT Exercise ADC_CC2 - Device Access Server
1.) Checking USB connections

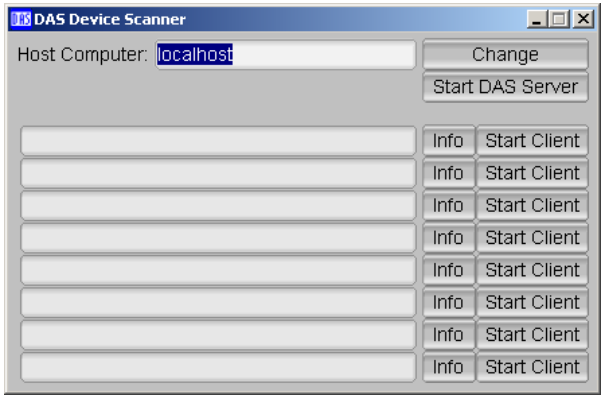


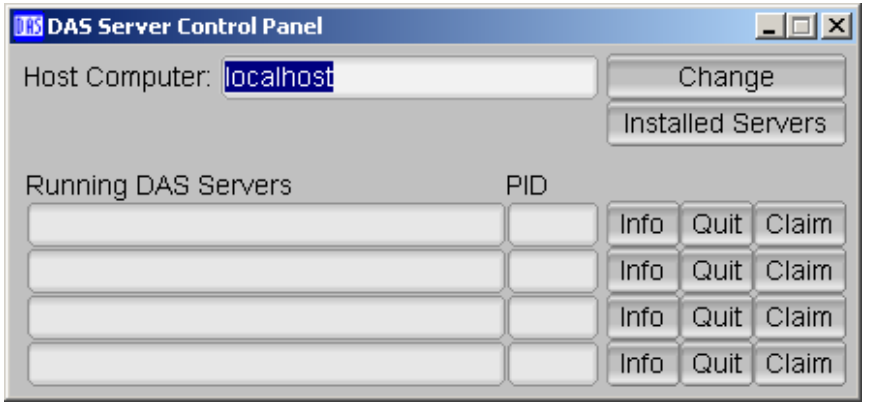
This gets identified only when COM port is used
- Via the USB interface on the Easykit with FTDI chip

The DAS JTAG composite device gets identified
- When miniWiggler is connected
- When USB Wiggler Box is connected
- Via the USB interface on the Easykits with FTDI chip

HOT Exercise ADC_CC2 - Device Access Server
2.) Check DAS status

1. Start DAS device scanner
2. Start DAS Server Control panel

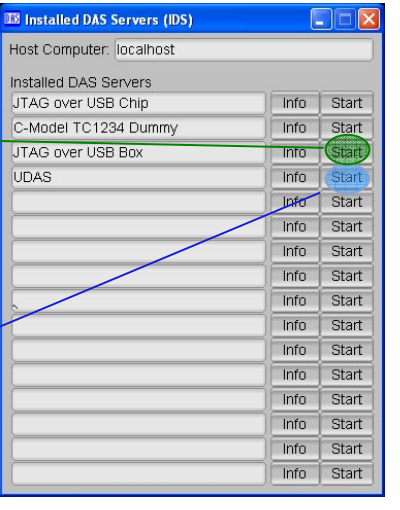




3. If DAS device scanner does not show any device, start the appropriate DAS server

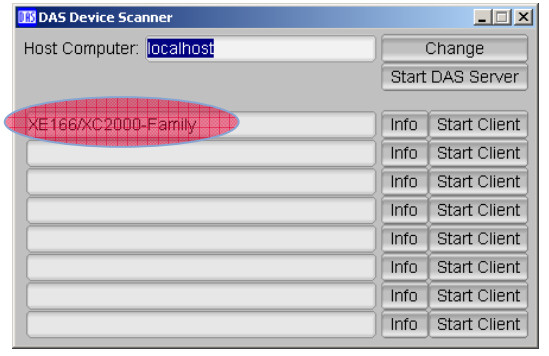
Incase you are connected via the USB Wiggler box, then start „JTAG over USB Box“

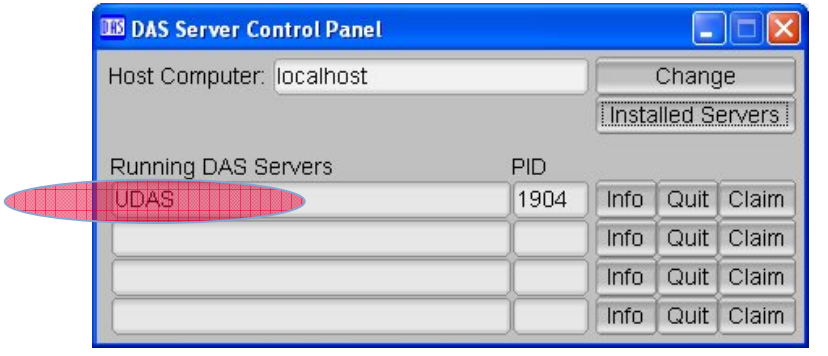
Incase you are connected via the FTDI chip or mini wiggler, then start „UDAS“



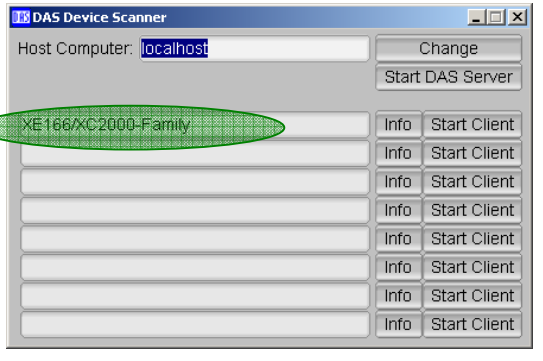
HOT Exercise ADC_CC2 - Device Access Server
3.) Starting the servers manually

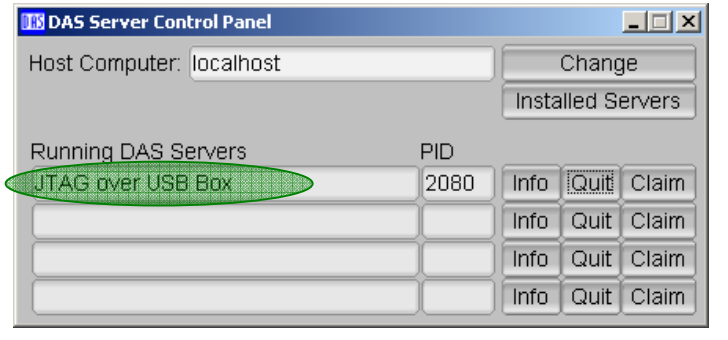
4. Incase „UDAS“ server is started and XC2000 easykit is connected via on-chip FTDI or via separate miniWiggler, following status changes could be noted





5. Incase „JTAG over USB Box“ server is started and XC2000 starter kit is connected via Wiggler box, following status changes could be noted

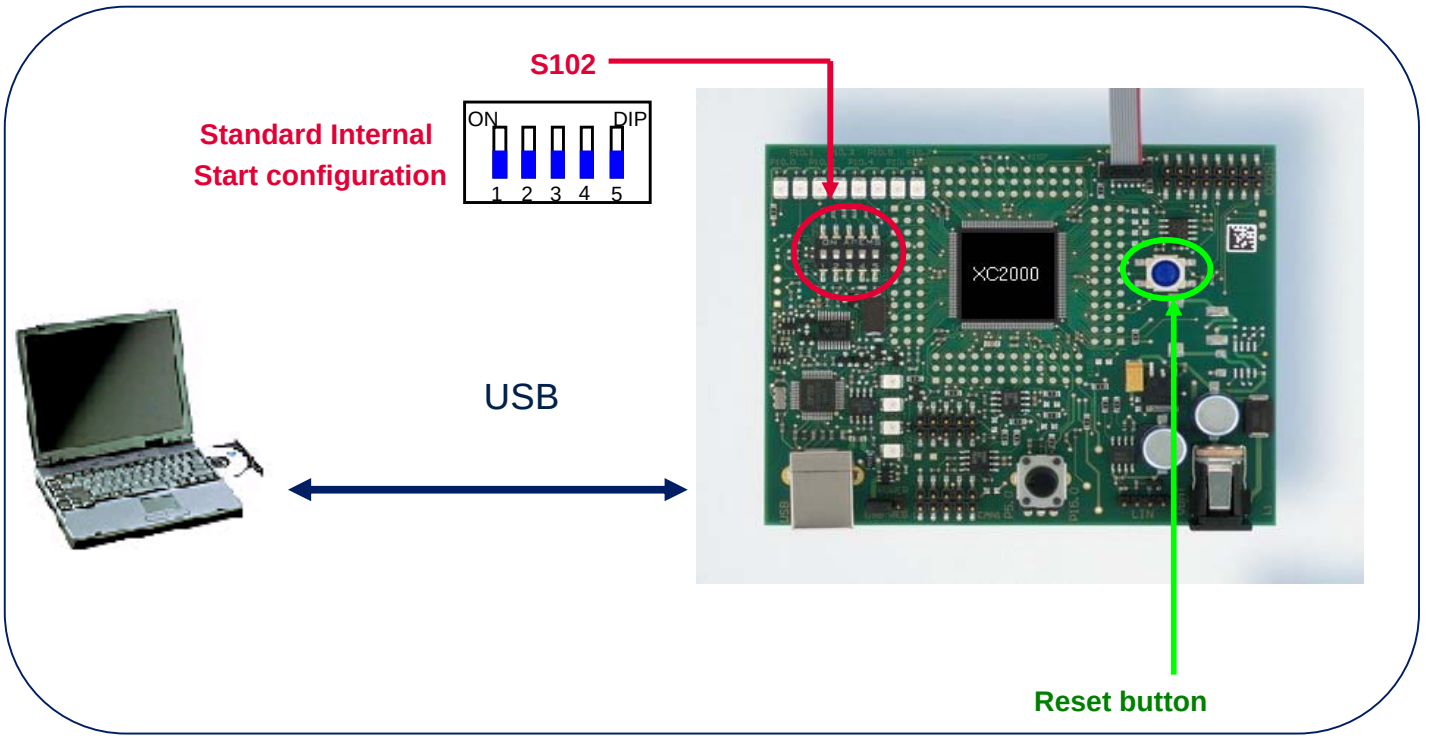




HOT Exercise ADC_CC2 – Tasking VX Toolset Connect XC2287M Easy Kit

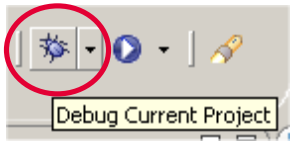
- Connect XC2287M Board to PC
- Modify The DIP Switch Settings, S102: **OFF-OFF-OFF-OFF-OFF**

  (Start from Internal Flash)
- Reset The Board (Press The Reset Button)

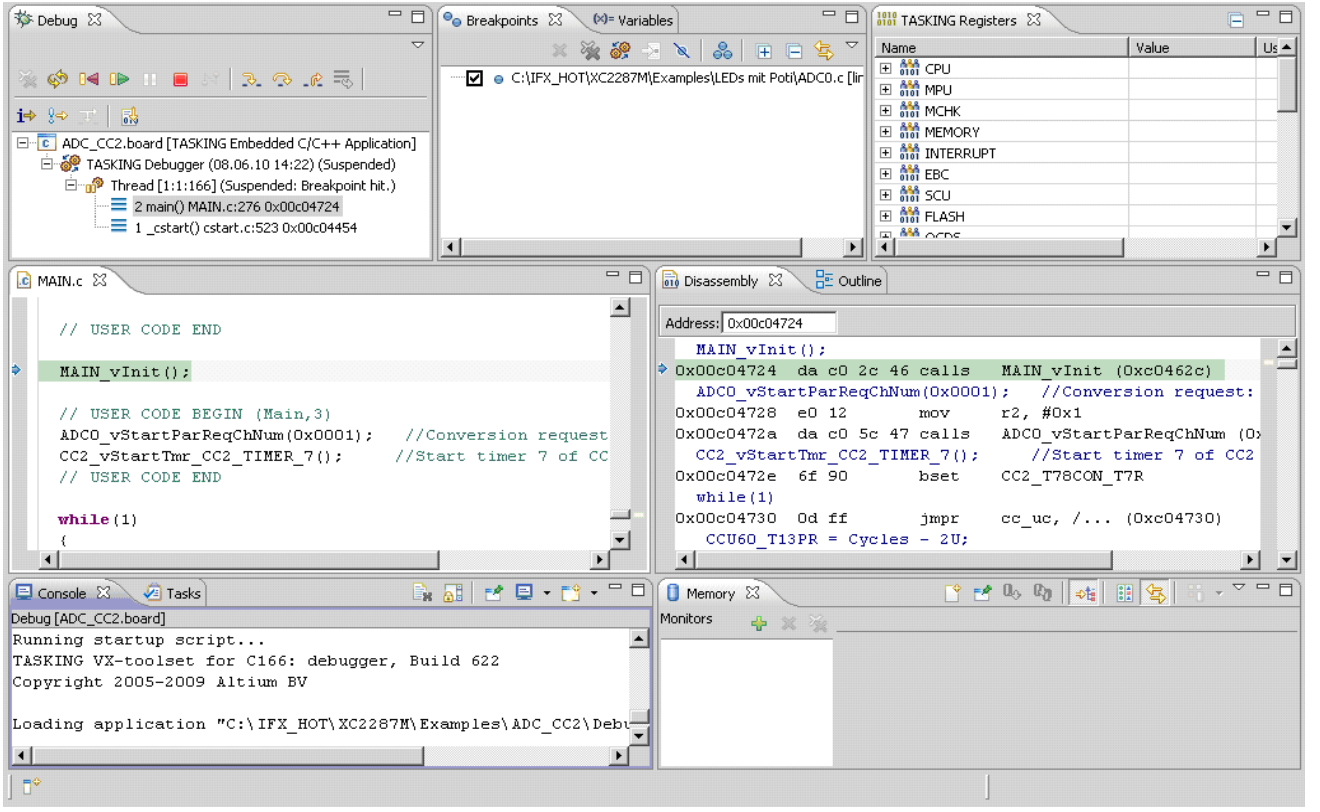


HOT Exercise ADC_CC2 – Tasking VX Toolset
Run Debugger

- Click on 1



- Click on ‘Resume’ and start program 2

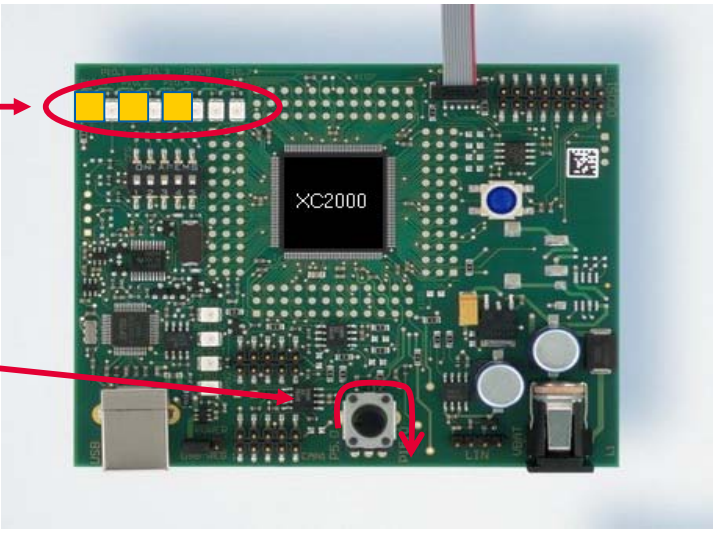


HOT Exercise ADC_CC2 – Tasking VX Toolset Run Debugger

- See Results
  - Now LED's Are Flashing in Different Pattern If The Input Voltage of Channel 0 of ADC0 Changes By Turning The Knob of The On-board Potentiometer

ADC Results on LED's

On-board Potentiometer



HOT Exercise ADC_CC2 – Tasking VX Toolset Run Debugger

- Verifications
  - Click on ‘Suspend’ 
  - Click on ‘Restart’ 
  - LED’s Stop Changing ON/OFF States
  - Click on ‘Resume’ 
  - LED’s display ADC value

Never stop thinking