XC2287M HOT

PEC Solution

Usage of PEC

Device: XC2287-104F
Compiler: Tasking Viper 2.2r1
Code Generator: DAvE 2.1
Usage of PEC

Let’s get started now!
XC2287M HOT Exercise PEC Interaction of Development Tools

IDE

Compiler
Assembler
Linker
Locator

Debugger
int main(){
    char a;
    long b;
    ...

Programming Tool

Debugger/Emulator

Target

DAvE

SW

HW
In this exercise you will:

- Configure the XC2287M with DAve
- Assign a potentiometer to an ADC channel
- Define ADC parameters
- Develop a program that will do ADC conversion continuously
- Configure PEC
- Do performance tests
Exercise goal:

Trigger ADC conversion by software

- The onboard potentiometer is connected to channel 0 (P5.0)
- A/D converter will run in software triggered autoscan mode
- Use wait for read mode
- Configure time base for performance measurements
- Configure PEC
HOT Exercise PEC - DAvE Configurations
Start DAvE

- **Start DAvE**
  - Click on the DAvE

- **Create a new project** (Startup Dialog pop up automatically)
  - Click on ‘Create a new project’ or select File -> New
  - Select microcontroller: ‘XC2287M’
HOT Exercise PEC - DAvE Configurations
Start DAvE (cont.)
HOT Exercise PEC - DAvE Configurations

- Project Settings
- Close the window

![Project Settings Window](image-url)
HOT Exercise PEC - DAvE Configurations
Save DAvE Project

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

1. Save project
2. Specihren in: PEC
3. Dateiname: PEC.dav
4. Dateityp: DAve project file (*.dav)

HOT Exercise PEC - DAve Configurations

Project Settings (cont.)
Click on ‘ADC Clock’
Configure ‘ADC Clock’

- Enable module
- default settings for others
- Close the windows by pressing
Click on ADC0
Configure ADC0 – General

- Arbitration Slot Functions – enable ‘arbitration slot 1’ only
- Arbitration Mode – enable ‘Arbitration started by pending conversion request’
- Others- default
HOT Exercise PEC - DAveE Configurations ADC Settings (cont.)

- Configure ADC0 – Channels
  - Click on ‘Configure Channel 0’
HOT Exercise PEC - DAvE Configurations
ADC Settings (cont.)

- Configure ADC0 – Channels
  - Enable Channel 0
  - Input Class – select ‘InputClass0’
  - Result Register - select ‘ResultReg0’
Configure ADC0 – Channels

Channel 0 is now selected
HOT Exercise PEC - DAvE Configurations
ADC Settings (cont.)

- Configure ADC0 – Parallel
  - Parallel Source Gating Configuration – select ‘EnabledAlways’
  - Autoscan Feature – enable ‘Autoscan’
Configure ADC0 – Result Register1

- Settings for Register 0 – enable interrupt SR0
- Settings for Register 0 – enable wait for read mode
Configure ADC0 – Interrupts

- Drag ‘ADC INT 0’ from Level 0 to Level 14, Group 0

Note: To change the level and the group of an interrupt source, click on it, drag it to its new position and drop it. To set an interrupt source to the non interrupting level (Level 0) click on it, drag it to the ‘Level 0’ list and drop it.
Configure ADC0 – Functions

- Click on ‘ADC_vInit’
- Click on ‘ADC0_vSetloadEvent’
- Click on ‘ADC0_vStartParReqChNum’
- Click on
Click on ‘GPT1’
Configure GPT1 – Module Clock

- Enable module

![Configure GPT1 - Module Clock](image.png)
Configure GPT1 – Timer 3

- Up/Down Control – select ‘Counter down’
- Timer Register (Underflow) – set as ‘100 µs’
- Interrupt Control – click on ‘Enable timer 3 interrupt’
Configure GPT1 – Interrupts

- Drag ‘GPT1 T3 INT’ from Level 0 to Level 15, Group 0
Configure GPT1 – Functions

- Initialization Function – click on ‘GPT1_cInit’
- Function Library (part 1) – click on ‘GPT1_vStartTmr’
- Function Library (part 1) – click on ‘GPT1_vStopTmr’
- Click on
Save your DAvE Project File

- Go to File → Save (or Save As) or click on
- Filename: “c:\IFX_HOT\XC2287M\Examples\PEC\PEC.dav”
Let DAvE Generate Code for You

- Go to **File → generate Code** or click on

- DAvE generated code files are
  - MAIN.C, MAIN.H
  - GPT1.C, GPT1.H
  - GPT2.C, GPT2.H
  - ADC0.C, ADC0.H
  - SCS.C, SCS.H
  - XC22XXREGS.H
Create New Work Space

- Click on [image]
- Filename: “c:\IFX_HOT\XC2287M\Examples”
- Click ‘OK’
Create New Project

Click on Workbench
HOT Exercise PEC – Tasking VX Toolset

1. Import DAVE Project

   - Click on File -> Import
   - Select Tasking VX-toolset for C166...
   - Click ‘OK’
**HOT Exercise PEC – Tasking VX Toolset**

- Import DAVE Project
  - Click `Infineon DAvE C166 Project´
  - Click ‘Next’
HOT Exercise PEC – Tasking VX Toolset

- Import DAvE Project
  - Add Dave Project ‘PEC’
  - Click ‘Finish’
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`
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:

```c
while(1)
{
    // USER CODE BEGIN (Main,4)
    BlinkLED();
    // USER CODE END
}
```
```c
void main(void)
{
    // USER CODE BEGIN (Main,2)
    // USER CODE END

    MAIN_vInit();

    // USER CODE BEGIN (Main,3)
    ADC0_vStartParReqChNum(0x0001); // Start conversion channel 1 to channel 0 / P5.0
    ADC0_vSetLoadEvent();           // Set software trigger
    GPT1_vStartTmr_GPT1_TIMER_3();
    // USER CODE END

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

    }
}
```
HOT Exercise PEC – Tasking VX Toolset
Add User Code – ADC0.C

//@Global Variables
//*******************************************************************
// USER CODE BEGIN (ADC0_General,7)
unsigned int size = 0;
unsigned int buffer[100] __at(0xA000);
// USER CODE END

_{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)
        buffer[size++] = (ADC0_RESRA0 << 4); // put ADC value in array
        // 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
    }

}}
// @Imported Global Variables
//*******************************************************************
// USER CODE BEGIN (GPT1_General,6)
extern int size;
extern int buffer[];
// USER CODE END

__interrupt(T3INT)  void GPT1_viTmr3(void)
{
    // USER CODE BEGIN (Tmr3,2)
    ADC0_CRMR1 = ADC0_CRMR1 && 0xFFEF;  //Disable autoscan
    GPT1_vStopTmr_GPT1_TIMER_3();
    // USER CODE END

    // USER CODE BEGIN (Tmr3,5)
    // USER CODE END

} // End of function GPT1_viTmr3
1. Click on ‘Build Project PEC’
Note: It is recommended to use the latest DAS version. Download the latest version at www.infineon.com\DAS
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 PEC - 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 PEC - 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 PEC – Tasking VX Toolset
Connect XC2287M Easy Kit

- Connect XC2287M Board to PC
- Modify The DIP Switch Settings, S102: OFF-OFF-OFF-OFF-OFF-OFF (Start from Internal Flash)
- Reset The Board (Press The Reset Button)
HOT Exercise PEC – Tasking VX Toolset
Run Debugger

1. Click on **Debug Current Project**.
2. Click on ‘Resume’ and start program.
HOT Exercise PEC – Tasking VX Toolset
Run Debugger

- See Results
  - Add Watch Expression
  - Add 'size' and 'buffer'
  - Click on 'Resume' and start program
See Results

- Click on ‘Suspend’

Result: 53 AD conversions in 100µs
HOT Exercise PEC - DAvE Configurations
ADC Settings (PEC)

- Click on ADC0

![Diagram of DAvE XC2287M (Release v2.0)]
Configure PEC – Functions

- Click on ‘ADC INT 0’, then on ‘Configure PEC Channel 0’
HOT Exercise PEC - DAveE Configurations
ADC Settings (PEC)

- Configure PEC channel 0
  - Click on ‘Continuous transfer’
  - Click on ‘Increment destination pointer’
  - Search for Source pointer / Destination pointer
  - Click on

![Diagram of DAveE configurations]

1. Continuous transfer
2. Increment destination pointer
3. Source pointer (ADC0_Result Register (ADC0_RESFO))
4. Destination pointer (User defined)
5. Click on the file icon
Save your DAvE Project File

- Go to **File → Save (or Save As)** or click on
- Filename: “c:\IFX_HOT\XC2287\Example\PEC\PEC.dav”
Let DAvE Generate Code for You

- Go to **File → generate Code** or click on

- DAvE generated code files are
  - MAIN.C, MAIN.H
  - GPT1.C, GPT1.H
  - ADC0.C, ADC0.H
  - SCS.C, SCS.H
  - XC22XXREGS.H

---

HOT Exercise PEC - DAvE Configurations Code Generation

Please read this document carefully and note the red-colored hints.

If you miss a file in the generated files list maybe you have forgotten to select the initialisation function of the related module.
Click on ‘Build Project PEC’
HOT Exercise PEC – Tasking VX Toolset
Run Debugger

1. Click on [Debug Current Project]

2. Click on ‘Resume’ and start program
HOT Exercise PEC – Tasking VX Toolset

Run Debugger

- See Results
- Click on ‘Suspend’

Result: 75 AD conversions in 100µs
Improvement: ~ 41 %
We commit.
We innovate.
We partner.
We create value.