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
Let's get started now!

Hardware Triggering of ADC
XC2287M HOT Exercise ADC_CC2
Interaction of Development Tools

IDE

Compiler
Assembler
Linker
Locator

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

Programming Tool

Target

Debugger/Emulator

DAvE

SW

HW
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
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 [DAvE]

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

Close the window
Save your DAvE project

- Path: C:\IFX_HOT\XC2287M\Examples\ADC_CC2

- Project name: ADC_CC2\ADC_CC2.dav
Save your DAve Project File

1. Open the DAve project.
2. Click on "Save Project As".
3. Enter the name "ADC_CC2" in the file name field.
4. Click on "Speichern" to save the project.
HOT Exercise ADC_CC2 - DAvE Configurations
ADC Settings

■ Click on ‘ADC Clock’
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

![Diagram of DAvE XC2287M (Release v2.0)]
Configure ADC0 – General

- Arbitration Slot Functions – enable ‘arbitration slot 1’ only
- Arbitration Mode – enable ‘Arbitration started by pending conversion request’

- Others - default
Configure ADC0 – Channels

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

- Enable Channel 0
- Input Class – select ‘InputCalsss0’
- Result Register - select ‘ResultReg0’
HOT Exercise ADC_CC2 - DAveE Configurations
ADC Settings (cont.)

- Configure ADC0 - Channels

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

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

- Parallel Source Gating Configuration – select ‘EnabledAlways’
- External Trigger for Parallel Source – select ‘CC2_17’
Configure ADC0 – Result Register1

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

- Drag ‘ADC INT 0’ from Level 0 to Level 15, Group 0
HOT Exercise ADC_CC2 - DAveE 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’
Configure CC2 – Module Clock

- Enable module
Configure CC2 – Timer 7

- Timer 7 Mode (T7M) – select ‘Timer mode’
- Timer 7 Register – set as ‘100,00ms’
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 - DAveE Configurations
CC2 Settings (cont.)

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

![Configuration screenshot]

1. Select CC17 allocation to timer T1
2. Set CC17 allocation to timer T0
3. Enter CC register (CC17) value: 0xD000
4. Confirm settings
Configure CC2 – Channels

- Now Channel 17 is selected
Configure CC2 – Functions

- Function Library (Part1) – configure ‘CC2_vStartTmr’

Click on

![Diagram of Capture/Compare Unit 2 (CAPCOM2)](image)

1. Functions
2. CC2_vStartTmr
3.
HOT Exercise ADC_CC2 - DAvE Configurations

Port Settings

- Click on ‘Port’
Configure – Ports

- Click on ‘Configure Port 10’
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

![Screenshot of configuration interface]

1. Initialization Function - click on ‘IO_vInit’
2. Click on [Port Setting]
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”
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
Create New Work Space

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

Click on Workbench
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’
Configure Target Board

1. Select the project in the navigator
2. Select ‘Project/Target Board Configuration’
3. Select ‘Infineon XC2000/XE166 Easykit Board’
4. Choose `XC2287M-104F´
5. 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
}
```
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
```c
_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
    }
}
```
Click on ‘Build Project ADC_CC2’
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

<table>
<thead>
<tr>
<th>Network adaptors</th>
<th>USB Composite Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universal Serial Bus controllers</td>
<td>Universal Serial Bus controllers</td>
</tr>
<tr>
<td>Secure USB Hub</td>
<td>Secure USB Hub</td>
</tr>
<tr>
<td>External USB</td>
<td>External USB</td>
</tr>
<tr>
<td>USB Composite Device</td>
<td>USB Composite Device</td>
</tr>
</tbody>
</table>

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

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

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

1. Click on

2. Click on ‘Resume’ and start program
- 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
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
We commit.
We innovate.
We partner.
We create value.