ADXL355 Accelerometer PMOD Demo

The **ADuCM360_demo_adxl355_pmdz** is an accelerometer demo project for the EVAL-ADICUP360 base board with the EVAL-ADXL355-PMDZ board, using the GNU ARM Eclipse Plug-ins in Eclipse environment.

General description

This project is an example for how to use EVAL-ADICUP360 board in combination with the EVAL-ADXL355-PMDZ accelerometer PMOD board.

The ADuCM360_demo_adxl355_pmdz project uses the EVAL-ADXL355-PMDZ which has the **ADXL355 3-axis MEMS accelerometer** on board.

The application reads the **X** , **Y** , and **Z** acceleration registers. The acceleration in the 3 axis is displayed in **[G]**. There is an internal temperature sensor in the ADXL355, which is converted by the on chip 12-bit ADC. The acceleration range can also be selected by setting the **ADXL355_RANGE** variable with the *ADXL355.h* file. (Values of [2, 4, and 8 are acceptable])

All the outputs are printed from the UART to the USER USB port and can be read on the PC using a serial terminal program, such as Putty or Tera Term.

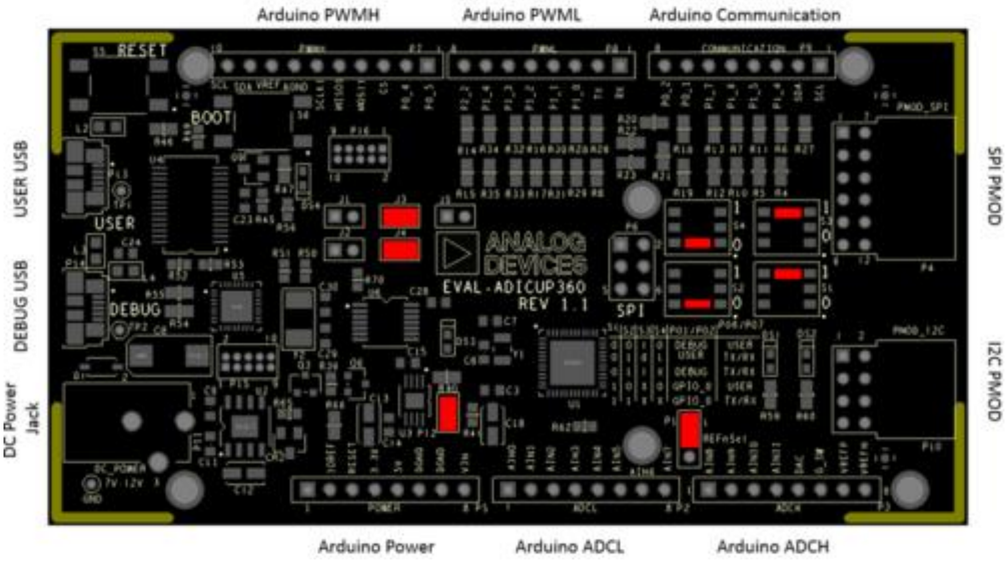
For precision applications, each **ADXL355** chip requires individual calibration which can be done by measuring and setting the definitions *ACC_TEMP_BIAS* and *ACC_TEMP_SENSITIVITY* parameters in the *ADXL362.h* file.

The temperature in degrees celsius, can be derived from the ADC readings **Tadc** using the predefined formula:

```
    Temp = (Tadc - ADXL355_TEMP_BIAS)/ ADXL355_TEMP_SLOPE) + 25;
```

Setting up the hardware

- To program the base board, set the jumpers as shown in the next figure. The important jumpers are highlighted in red.



- Plug the EVAL-ADXL355-PMDZ board in the EVAL-ADICUP360 base board, via the PMOD_SPI port (P4).
- Power EVAL-ADICUP360 base board via the Debug USB.(If programmed, switch to the USER USB for serial terminal display)

Obtaining the source code

We recommend not opening the project directly, but rather import it into Eclipse and make a local copy in your Eclipse workspace.

To learn how to import the **ADuCM360_demo_adxl355_pmdz** project form the projects examples in the Git repository, please click on How to import existing projects from the GIT Repository.

The source code and include files of the **ADuCM360_demo_pmodacl2** can be found on Github:

AduCM360_demo_adxl355_pmdz at Github

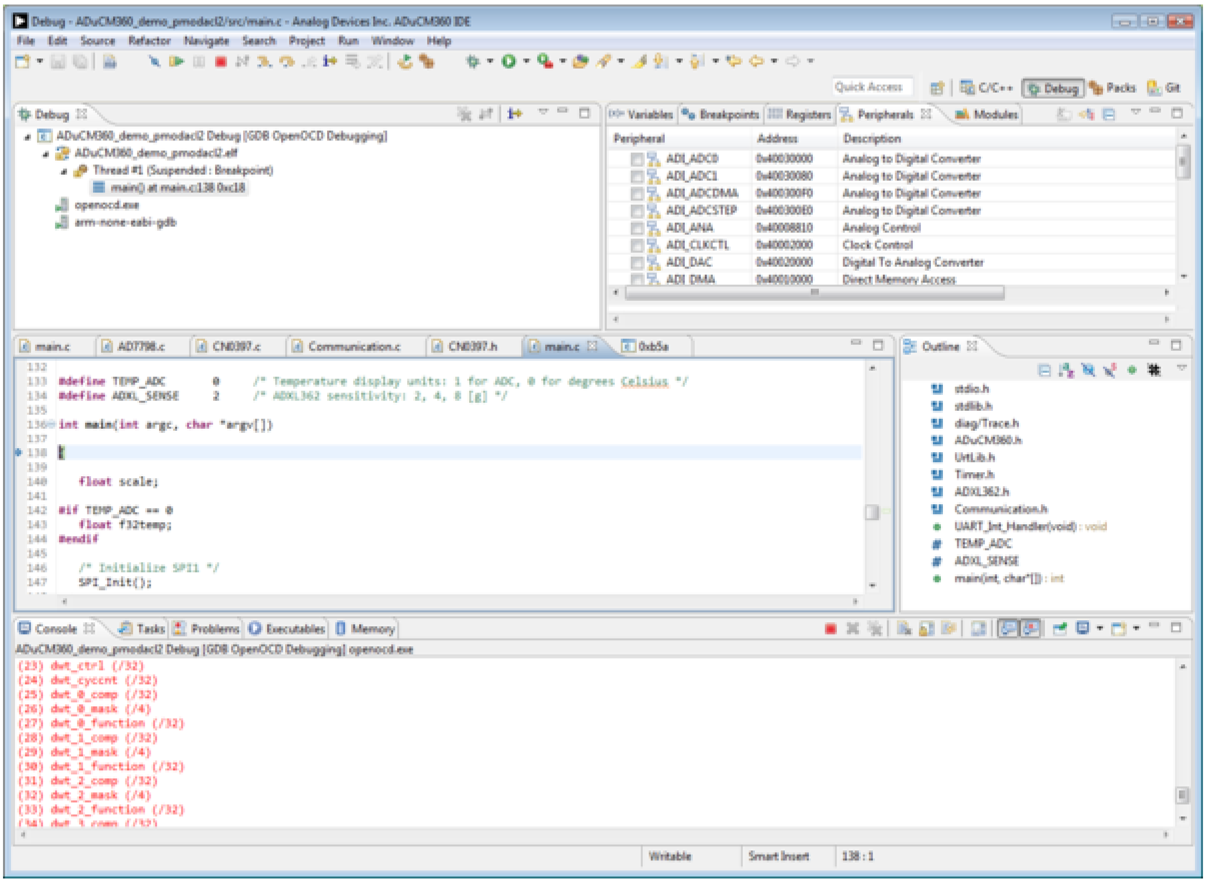
Importing the ADuCM360_demo_adxl355_pmdz project

The necessary instructions on how to import **ADuCM360_demo_adxl355_pmdz** project in your workspace can be found in the section, Import a project into workspace.

Debugging the ADuCM360_demo_adxl355_pmdz project

- A debug configuration must be set up for this project in order to have the possibility to program and to debug the **ADuCM360_demo_adxl355_pmdz** project. To do this, follow the instructions from Setting up a Debug Configuration Page.

- Make sure the target board is connected to the computer (via **DEBUG USB**) and using the tool bar, navigate to the small Debug icon🐞▾and select the debugging session you created. The application will programmed and the program execution will stop at the beginning of the main() function.



```
ADuCM360_demo_pmodacl2 Debug [GDB OpenOCD Debugging]
  ADuCM360_demo_pmodacl2.elf
    Thread #1 (Suspended : Breakpoint)
      main() at main.c:138 0xc18
  openocd.exe
  arm-none-eabi-gdb
```

Peripheral	Address	Description
ADI_ADC0	0x40030000	Analog to Digital Converter
ADI_ADC1	0x40030080	Analog to Digital Converter
ADI_ADCDMA	0x400300F0	Analog to Digital Converter
ADI_ADCSTEP	0x400300E0	Analog to Digital Converter
ADI_ANA	0x40008810	Analog Control
ADI_CLKCTL	0x40002000	Clock Control
ADI_DAC	0x40020000	Digital To Analog Converter
ADI_DMA	0x40010000	Direct Memory Access

```

#define TEMP_ADC        0      /* Temperature display units: 1 for ADC, 0 for degrees Celsius */
#define ADXL_SENSE      2      /* ADXL362 sensitivity: 2, 4, 8 [g] */

int main(int argc, char *argv[])

{

   float scale;

#if TEMP_ADC == 0
   float f32temp;
#endif

   /* Initialize SPI1 */
   SPI_Init();
```

```
stdio.h
stdlib.h
diag/Trace.h
ADuCM360.h
UrtLib.h
Timer.h
ADXL362.h
Communication.h
UART_Int_Handler(void) : void
TEMP_ADC
ADXL_SENSE
main(int, char*[]) : int
```

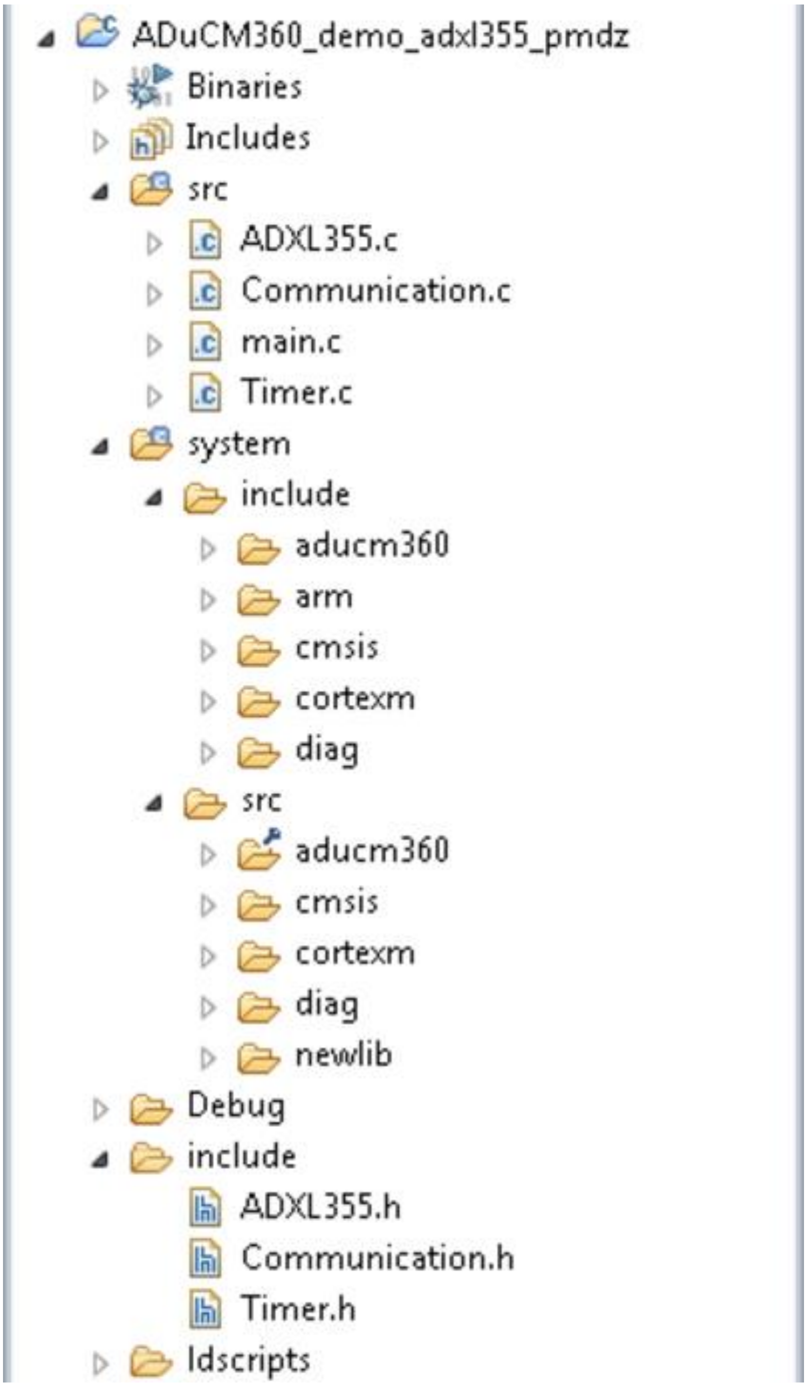
```
ADuCM360_demo_pmodacl2 Debug [GDB OpenOCD Debugging] openocd.exe
(23) dwt_ctrl (/32)
(24) dwt_cyccnt (/32)
(25) dwt_0_comp (/32)
(26) dwt_0_mask (/4)
(27) dwt_0_function (/32)
(28) dwt_1_comp (/32)
(29) dwt_1_mask (/4)
(30) dwt_1_function (/32)
(31) dwt_2_comp (/32)
(32) dwt_2_mask (/4)
(33) dwt_2_function (/32)
(34) dwt_3_comp (/32)
```

- Use step-by-step execution or directly run the program.

After completion of the steps above the program will remain written into the system flash and it will run by default every time the board is powered up.

Project structure

The **ADuCM360_demo_adxl355_pmdz** project use basic ARM Cortex-M C/C++ Project structure. This project contains: system initialization part - disabling watchdog, setting system clock, enabling clock for peripheral; port configuration for SPI0, accelerometer sensor use; SPI read/write functions; sensor monitoring.



```
ADuCM360_demo_adxl355_pmdz
    Binaries
    Includes
    src
        ADXL355.c
        Communication.c
        main.c
        Timer.c
    system
        include
            aducm360
            arm
            cmsis
            cortexm
            diag
        src
            aducm360
            cmsis
            cortexm
            diag
            newlib
    Debug
    include
        ADXL355.h
        Communication.h
        Timer.h
    ldscripts
```

In the **src** and **include** folders you will find the source and header files related to ADXL355 application. You can modify as you wanted those files. The *Communication.c/h* files contain SPI and UART specific data, meanwhile the *ADXL355.c/h* files contain the accelerometer data. Here are parameters you can configure:

- **Temperature sensor calibration values-** *ADXL355_TEMP_BIAS* and *ADXL355_TEMP_SLOPE* paramaters - find your values based on the calculation formula (*ADXL355.h*):

```
#define ADXL355_TEMP_BIAS           (float)1852.0
```

```
#define ADXL355_TEMP_SLOPE          (float)-9.05
```

- **Accelerometer range setting** - *ADXL_RANGE* parameter - 2, 4, or 8 are acceptable values to set the [g] range for the ADXL355 (*ADXL355.h*).

```
#define ADXL_SENSE        2
```

- **Sensor activity and inactivity thresholds** - *ACT_VALUE* and *INACT_VALUE* paramaters used to determine at which acceleration values the sensor can react at sleep/wake-up commands (*ADXL355.h*):

```
#define ACT_VALUE          50
#define INACT_VALUE        50
```

- **Sensor activity and inactivity time** - *ACT_TIMER* and *INACT_TIMER* paramaters used to determine sleep/wake-up intervals(*ADXL355.h*):

```
#define ACT_TIMER          50
#define INACT_TIMER        50
```

The **system** folder contains system related files (try not to change these files):

- **ADuCM360** – contains low levels drivers for ADuCM360 microcontroller.
- **CMSIS** – contains files related to ADuCM360 platform, such as: *ADuCM360.h* (registers definitions), *system_ADuCM360.c/h* (system clock), *vectors_ADuCM360.c* (interrupt vector table).
- **cortexm** – contains files for system management (start-up, reset, exception handler).