Motion Driver 6.1 – Getting Started Guide

Table of Contents

1 Revision History

Revision Date	Revision	Description
06/27/2014	1.0	Initial Release
07/17/2014	1.1	Added STM32F4 information

2 Purpose

Motion Driver is an embedded software stack of the sensor driver layer that easily configures and leverages many of the features of the InvenSense motion tracking solutions. The motion devices supported are MPU6050/MPU6500/MPU9150/MPU9250. Many of the features of the hardware and the on board Digital Motion Processor (DMP) are encapsulated into modular APIs which can be used and referenced.

Motion Driver is designed as a solution which can be easily ported to most MCUs. With the release of the Motion Driver 6.0 it includes a 9-axis solution for ARM MCUs and the TI-MSP430. 6-axis only solutions should continue to reference the Motion Driver 5.1.2 for easier understanding of the software.

This document details how to set up the hardware and get the default projects up and running. It is recommended as a good way to understand the Motion Driver algorithms, DMP, and MPU HW features.

3 Release Package

MD6.0 release package contains example projects of the TI-MSP430 using Code Composer as well as STM32F4 and STM32L using IAR. It also contains binary MPL libraries for 9-axis fusion precompiled for ARM processors and TI-MSP430 processors. MPL libraries for arm uses gcc 4.7.2 compiler.

- **…\arm\STM32F4_MD6 :** Directory which contains the IAR project for STM32F4 Discovery Evaluation Board and the InvenSense motion solution. The STM32F4 is a Cortex-M4 MCU core. The IAR project file is located under .\STM3F4L_MD6\Projects\eMD6\EWARM\STM32F4_MD6.eww

- **…\arm\STM32L_MD6 :** Directory which contains the IAR project for STM32L Discovery Evaluation Board and the InvenSense motion solution. The IAR project file is located under .\STM32L_MD6\Projects\eMD6\EWARM\STM32L_MD6.eww

- **...\documentation :** All relevant documentations regarding MD6 is under this directory

- **…\eMPL-pythonclient :** Python client used to test and demonstrate the motion device performance as well as display log information

- **…\mpl libraries :** Directory which contains the InvenSense Proprietary binary MPL (Motion Processing Library) used in the MD6.0. ARM libraries are compiled using GCC 4.7.2 while the TI libraries are using Code Composer 5.5

- **…\msp430\eMD-6.0 :** Contains the Code Composer project for MD6.0.

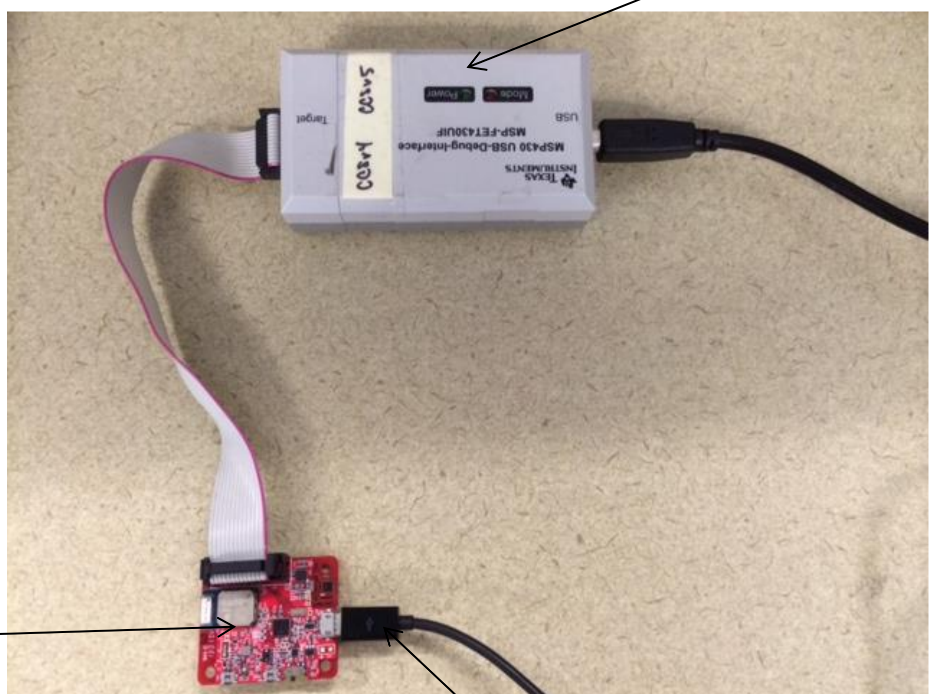
4 Starting with the TI-MSP430 Project

4.1 Requirements

- Code Composer Studio to compile MSP430 example
- TI-MSP430 JTAG for downloading and debugging
- Motion Driver 6.0 source files
- InvenSense CA-SDK evaluation board (can be purchased through invensense.com)

4.2 Connecting the Hardware

TI-MSP430 JTAG – connect to PC with Code Composer Software and also to the CA-SDK.

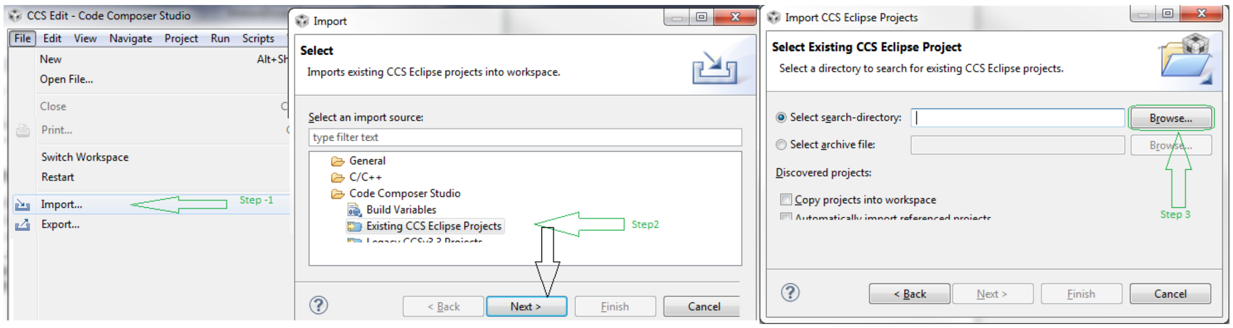


InvenSense CA-SDK evaluation board

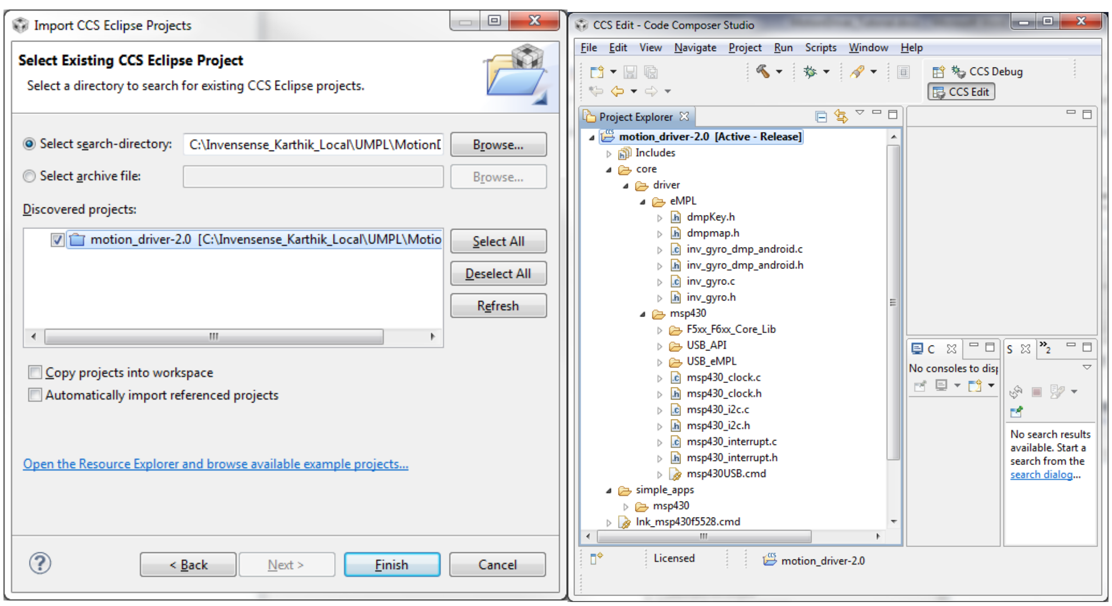
Micro-USB – Connect to PC for power and CA-SDK output

4.3 Opening and Compiling the MSP430 Project

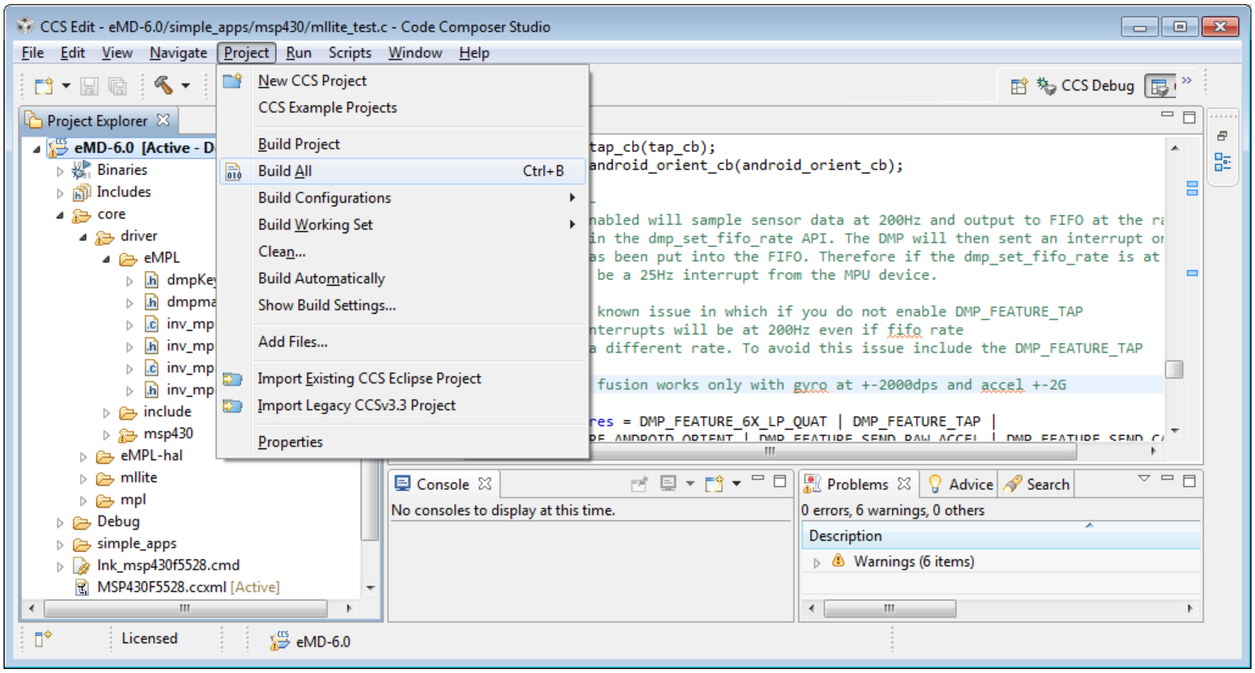
- Select import under the file menu.
- Choose existing CCS eclipse project.
- Click the browse button to select the Motion Driver folder.



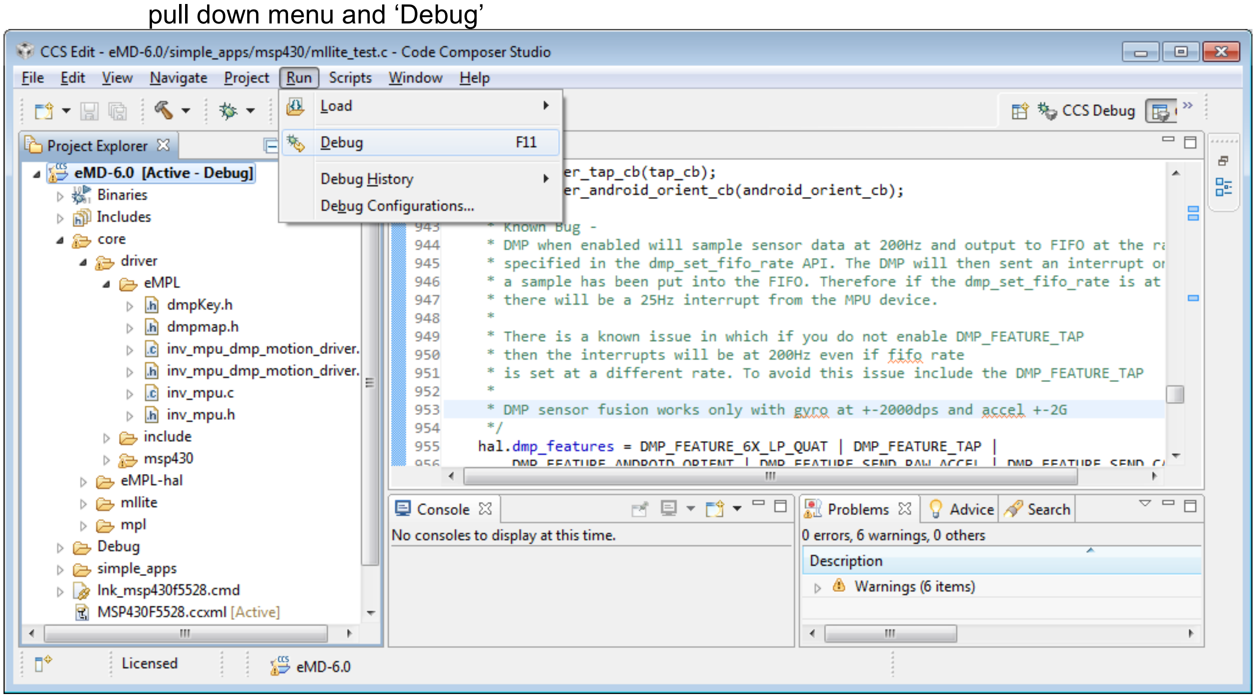
- Open the Motion Driver project by clicking finish.



- Select the ‘Project’ pull-down menu and ‘Build All’ to compile the project



- With the JTAG and CA-SDK hardware connected download the firmware by selecting the ‘Run’ pull down menu and ‘Debug’



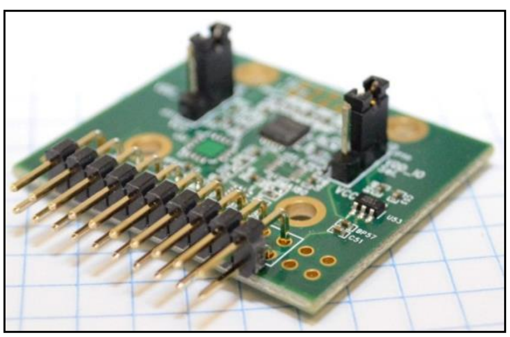
- You can then run the firmware through the debugger or turn off and on the CA-SDK to run off flash.

5 STM32L (Cortex-M3) Discovery Board Project

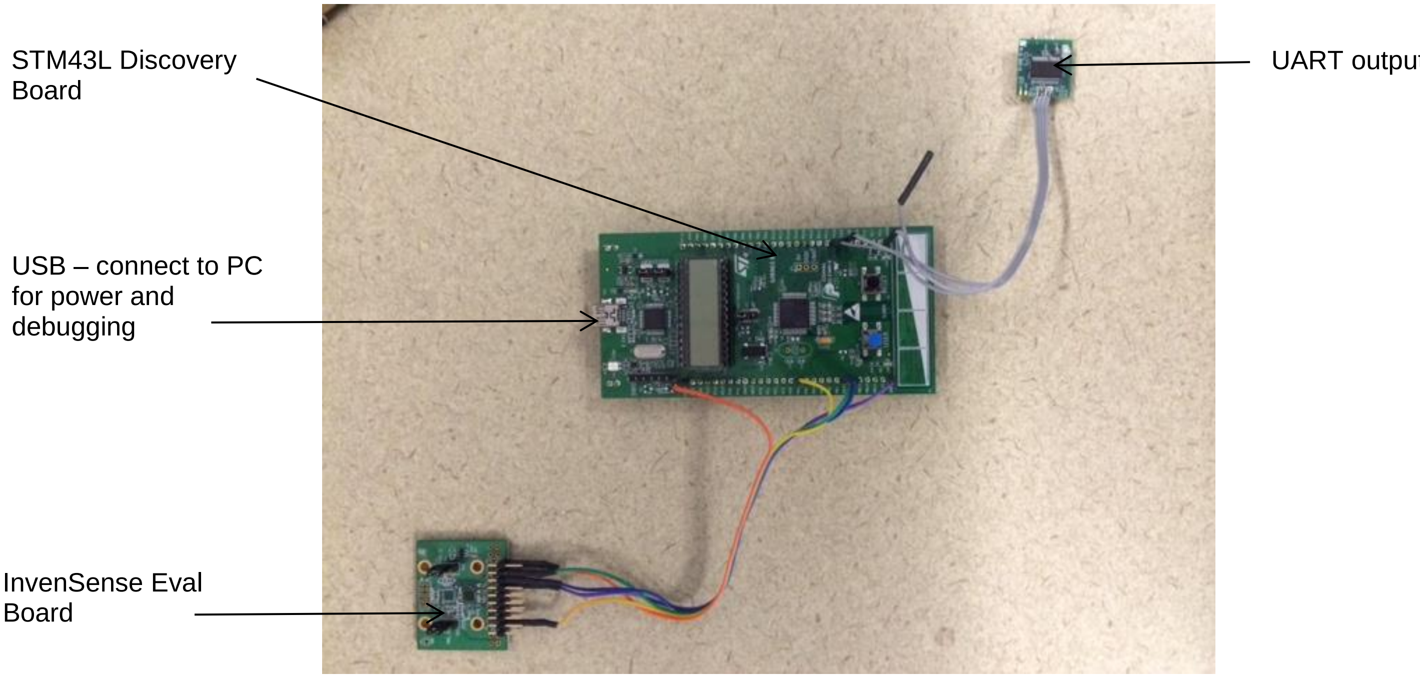
5.1 Requirements

- IAR ARM Workbench Compiler
- STM32L Discovery Evaluation Board (purchasable through DigiKey, Mouser, etc…)

- Motion Driver 6.0 source files
- InvenSense evaluation boards for MPU6050 or MPU6500 or MPU9150 or MPU9250

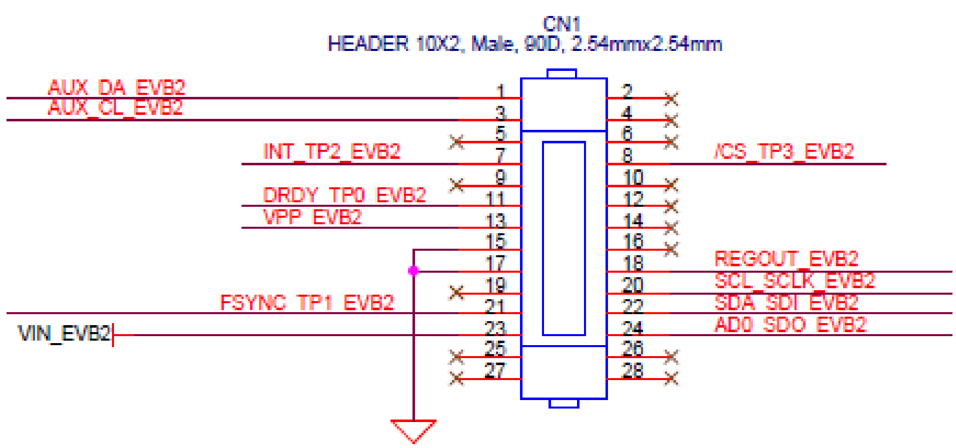


5.2 Connecting the Hardware



- **InvenSense Eval Board connection to Discovery Board**

The connection from the InvenSense eval board to the discovery board will require wiring between the two PCB boards. The InvenSense eval board pin outs are similar



To connect to the Discovery Board you will need to connect these 5 pins

EVB Header Pin Number	Description	Discovery Board GPIO Pin Number
3	INT output	PA1
13	GND	GND
19	VCC_IN	EXT_3V
16	I²C SCL	PB10
18	I²C SDA	PB11

- **Discovery Board UART Output**

The MD6.0 outputs via data via it's UART1 pins. The data is used by the python client to display information for the user. The pins are

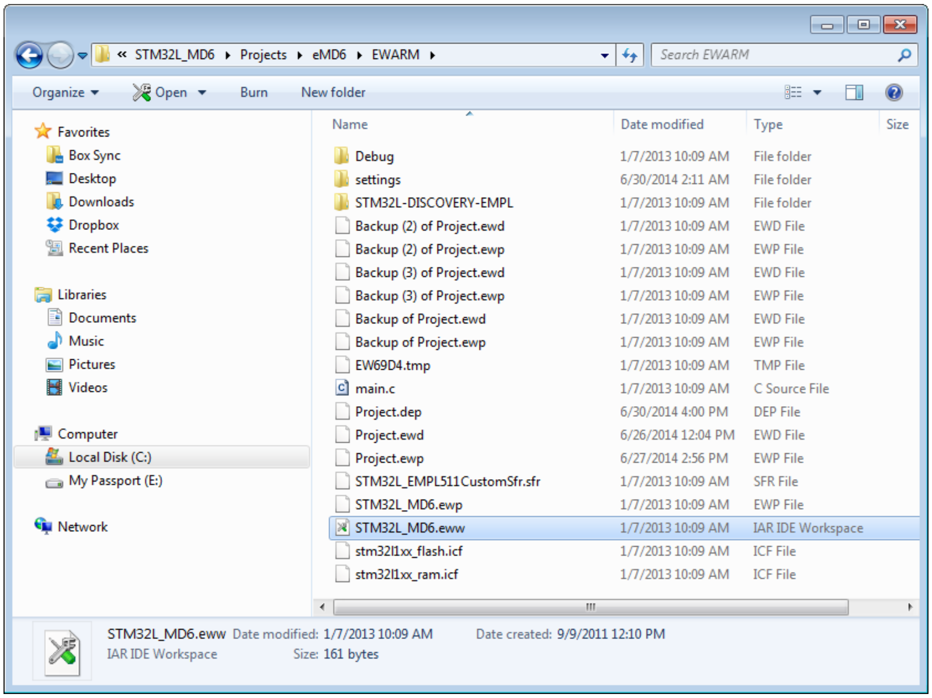
Discovery Board UART Out Pin Number	Description
PA9	UART Tx
PA10	UART Rx

You will need to use a UART convertor to the PC. There are several UART to Serial or UART to USB convertors available.

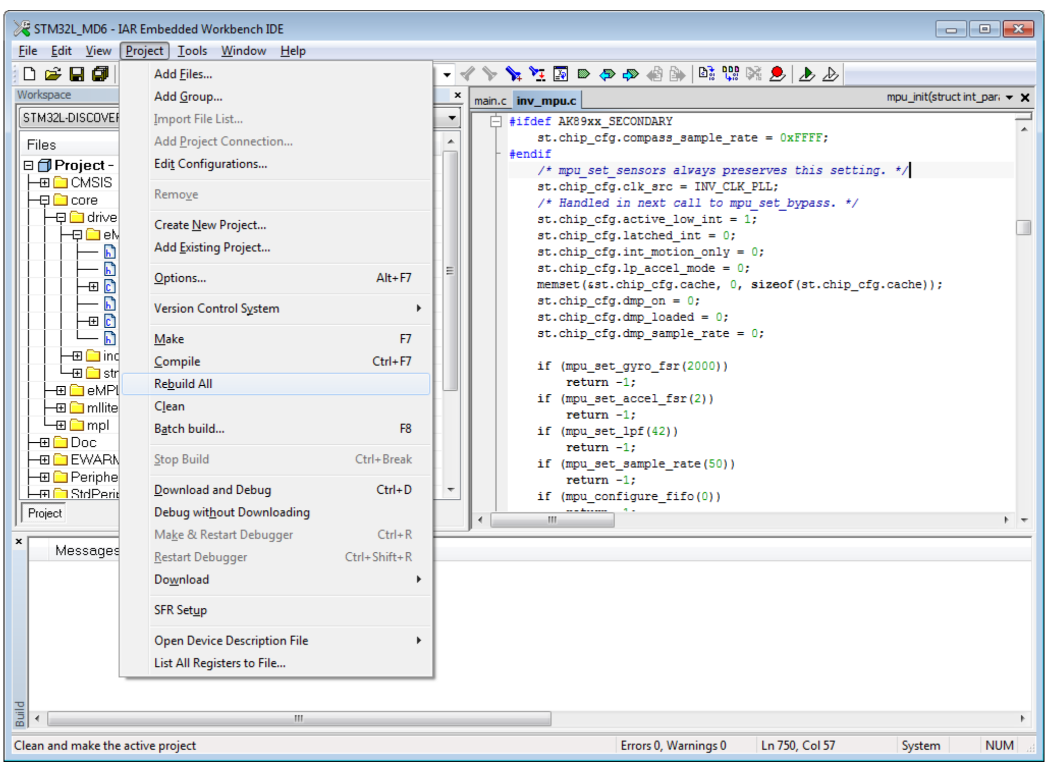
5.3 Opening and Compiling the IAR Project

- Double click on the IAR ARM project file to automatically open the workspace in IAR ARM compiler. Project file is under the directory

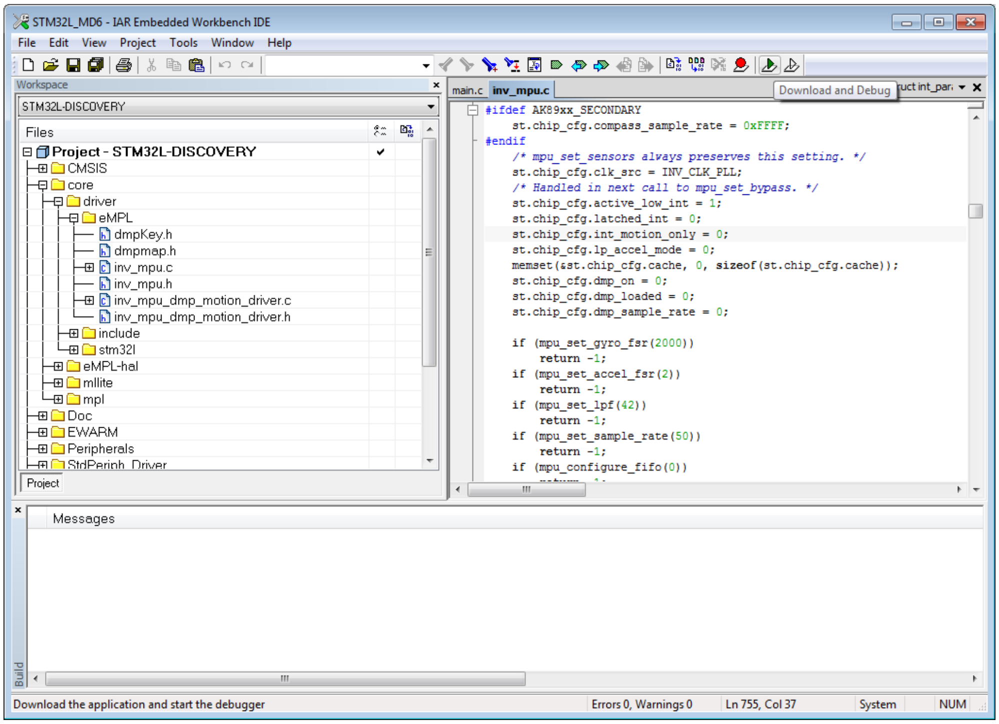
  ../STM32L_MD6/Projects/eMD6/EWARM/STM32L_MD6.eww



- Select the 'Project' pull down menu and 'Rebuild All'



- With the hardware connected, hit the 'Download and Debug' ICON



6 STM32F4 (Cortex-M4) Discovery Board Project

The STM32F4 (Cortex-M4) port is similar to the STM32L port with the following differences

- Project compiled for STM32F4-Discovery Board using board support files for the STM32F407VG. The Discovery Board is purchasable through the usual distribution channels

- MPL library compiled in arm gcc compiler 4.7.2 but with Cortex-M4 specific settings for better optimization. It can still work with the MPL library for generic ARM core.
- UART output from STM32F-Discovery Board

Discovery Board UART Out Pin Number	Description
PA2	UART Tx
PA3	UART Rx

Most of the software is the same except for the system and board related files.

7 Python Client

A python client is included with the release package to test the performance and display log information. The client can be found in the release package under the directory

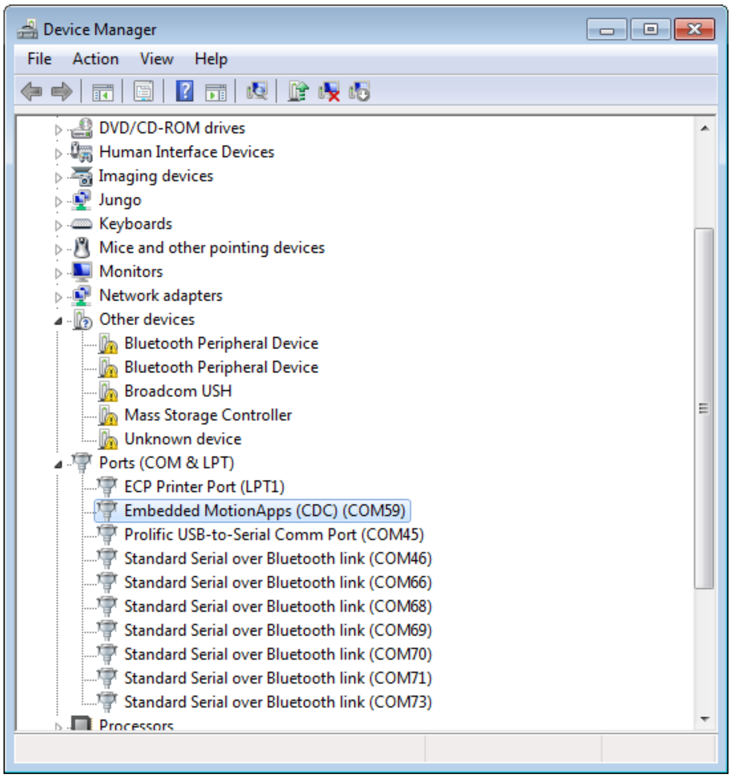
..\eMPL-pythonclient\

The python client also accepts user input and provides the input the sample HAL Application. The user would be able to enable/disable sensors, enable computation algorithms, enable hardware features, and view log information. You would need to install Python (version 2.5 and above), pyserial and pygame for the python script to execute.

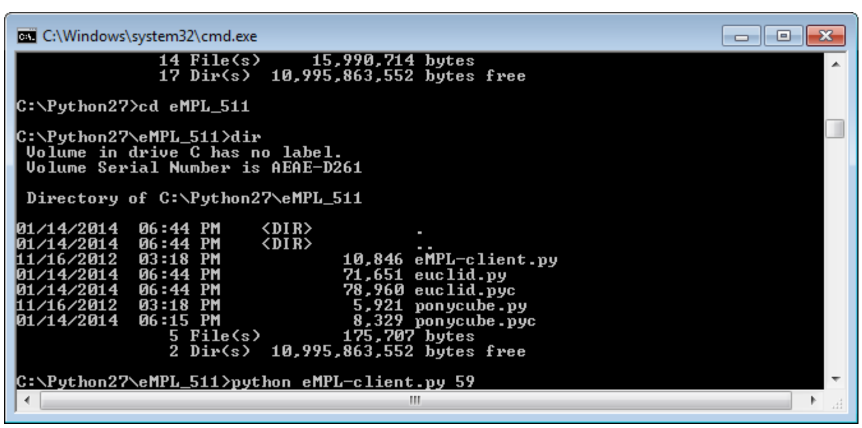
- Installing Python 2.7 (32-bits version) or above, pyserial, and pygame

  Python: https://www.python.org/downloads/
  Pyserial: https://pypi.python.org/pypi/pyserial
  Pygame: http://www.pygame.org/download.shtml

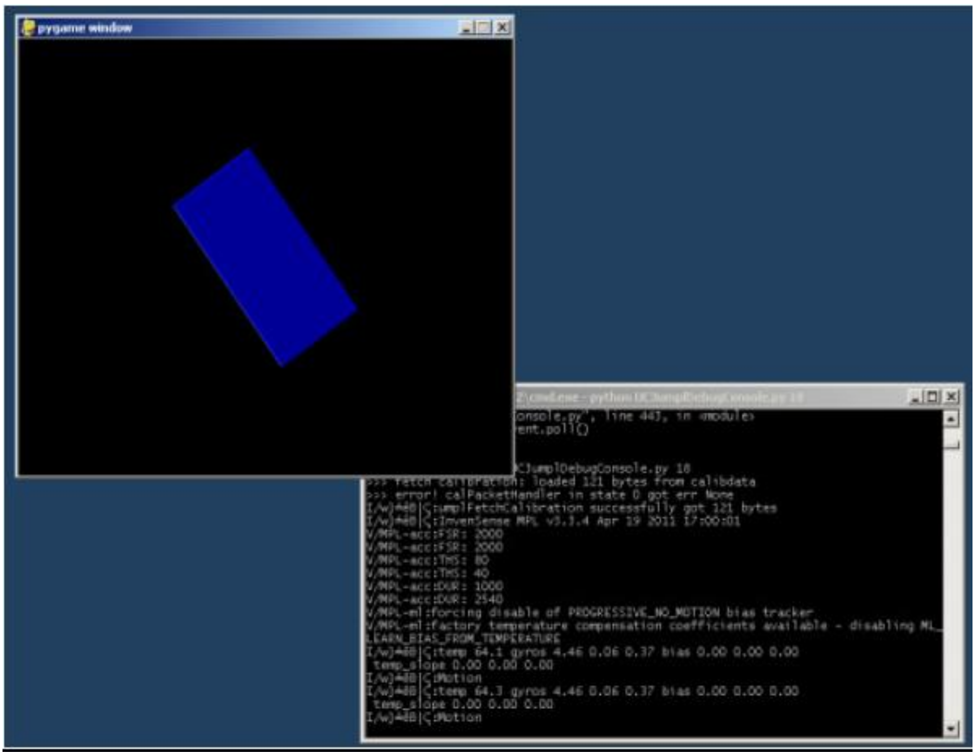
- Connect your flashed and working hardware to your PC and find the COM port in the device manager if the connected device



- Start the python client by opening up a command prompt window and browse to the python client directly and enter the following command
  - python eMPL-client.py <COM PORT NUMBER>



- 2 Windows will pop one. 1 contains a 3D Cube which corresponds with the quaternion angles outputted from the device. The other window will display any related logs or data



- Motion Driver 6.0 can accept input commands and display various different data. You must first make sure the cube window is the focused window then type in the input command.

  For TI-MSP430 you must first type 'inv' then the command.

  For IAR project you only need to type the command only.

  Commands 'l', 's', and 'x' are for MSP430 only.

  - '8' : Toggles Accel Sensor
  - '9' : Toggles Gyro Sensor
  - '0' : Toggles Compass Sensor
  - 'a' : Prints Accel Data
  - 'g' : Prints Gyro Data
  - 'c' : Prints Compass Data
  - 'e' : Prints Eular Data in radius
  - 'r' : Prints Rotational Matrix Data
  - 'q' : Prints Quaternions

- ‘h’ : Prints Heading Data in degrees
- ‘i’ : Prints Linear Acceleration data
- ‘w’ : Get compass accuracy and status
- ‘d’ : Register Dump
- ‘p’ : Turn on Low Power Accel Mode at 20Hz sampling
- ‘l’ : Load calibration data from flash memory
- ‘s’ : Save calibration data to flash memory
- ‘t’ : run factory self test and calibration routine
- ‘1’ : Change sensor output data rate to 10Hz
- ‘2’ : Change sensor output data rate to 20Hz
- ‘3’ : Change sensor output data rate to 40Hz
- ‘4’ : Change sensor output data rate to 50Hz
- ‘5’ : Change sensor output data rate to 100Hz
- ‘,’ : set interrupts to DMP gestures only
- ‘.’ : set interrupts to DMP data ready
- ‘6’ : Print Pedometer data
- ‘7’ : Reset Pedometer data
- ‘f’ : Toggle DMP on/off
- ‘m’ : Enter Low Power Interrupt Mode
- ‘x’ : Reset the MSP430
- ‘v’ : Toggle DMP Low Power Quaternion Generation