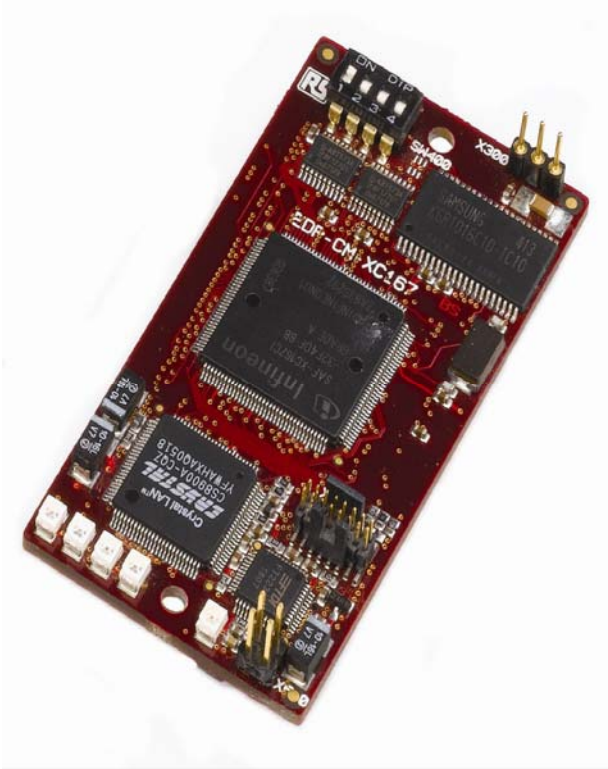
Embedded Development Platform

Getting Started Guide for XC167 Command Module

EDP-CM-XC167



Version 3.11
February 2011

RS EDP
EMBEDDED DEVELOPMENT PLATFORM

Contents

1. Introduction

To get the most out of the EDP platform it's important to understand the concept of the EDP system correctly. This is detailed in the user manual for the Base Boards which can be downloaded as a pdf file **RS EDP-BB-SystemBaseBoard User Manual Vx**, from the RS EDP website.

The base boards come in both 2 position and 4 position formats and share a common user manual. Please read this manual to get an understanding of the system.

Each of the Command Modules (CM) and Application Modules (AM) has its own user manual, so again these documents must be read to get an understanding on how to use the modules.

Each of the boards comes with its own suite of software to fully exercise the EDP Application Modules and the peripherals available on the MCU device.

An EDP system usually consists of one Command Module and one or more Applications Modules plugged into a Base Board. A minimum system just has a Command module and Base Board, for example a simple web server operating through an Ethernet connection.

The Command Module dictates whether the whole system uses a supply voltage of +3.3V or +5.0V. This particular CM module use a +3.3V microcontroller (MCU) and so the board is configured as such. The user can check the Vcc_CM signal on the Base Board break-out header to confirm the system voltage.

There are 144 pins on the MCU and these are connected via various link options to the Base Board. The Base Board then routes these signals to the Application Modules thereby allowing the CM Module to communicate with the Application Modules.

As many of the MCU pins have more than one function it can make the mapping of the connections rather complex so there are additional support documents available to help you with this. The first is the Pin Allocation Spread Sheet. One spread sheet is available for each of the CM Modules. The one for the XC167 module is called:

Pin Allocation - 144 pin XC167 Command Module Rev xx

This spreadsheet also forms part of the User Manual for the XC167 CM module. It details which pins are mapped to the Base Board backplane and the various link options which need to be configured to connect them accordingly.

To get an appreciation of how the Application Modules are mapped to the backplane and how the CPU Module can connect to them, a Mapping Aid exists. The one for the XC167 module is called:

Mapping Aid RS-EDP-XC167 Rev xx

This mapping aid also forms part of the User Manual for the XC167 and at a glance you can see what resources are required to get the best out of each Application Module.

Other useful documents you will need are the circuit diagrams for the modules you wish to use. These are contained in the back of each user manual.

So before you start to use the RS EDP system make sure you have to hand the following documents:

- Base Board User Manual
- Appropriate CM module User Manual
- Application Module User Manuals (as required)

2. Prepare to run the 'Hello World' Program

Program development is performed on a PC running suitable software linked to a USB debug connection on the EDP Base Board.

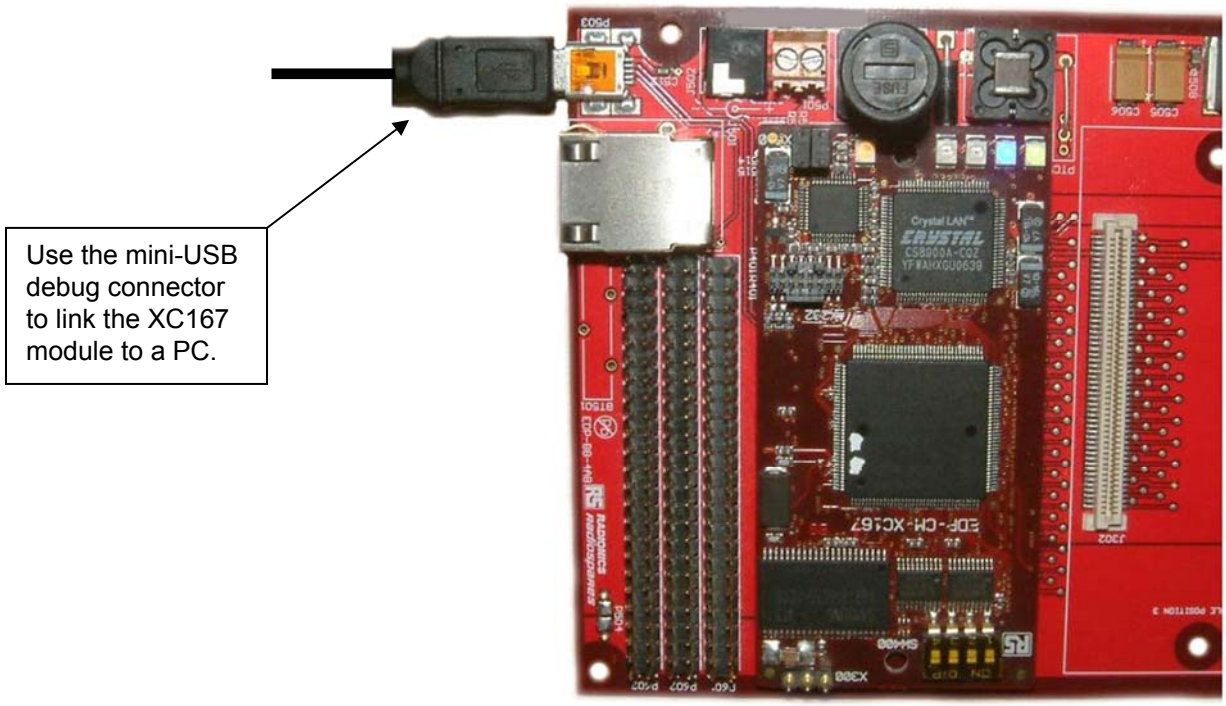
2.1 Software requirements

To build, download and run your first program you will need to have a suitable Integrated Development Environment (IDE) running on a PC. If necessary download **Keil µVision 4** from the Keil website. This is a full-featured free trial version that is limited to 32Kbytes of output code. You can upgrade to the full version later or try one of the alternatives discussed in an appendix to the User Manual. The IDE handles all aspects of code production as it contains a source code editor, assembler, C compiler, Flash programmer and debug tools.

Also required is the **DAS Tool Interface** driver which can be downloaded free from the Infineon web site. The Keil IDE always defaults to the ULINK driver, so at the start of a new Project the Debug/Programming settings will need to be switched to the DAS driver (See below).

2.2 Hardware requirements

On the XC167 CM board is an FTDI chip which provides a JTAG to USB interface. The USB end of this interface is bought down onto the Base Board and onto the mini-USB debug connector. This mini-USB allows the user to connect to a host PC and to run a debug session, including programming of the Flash memory, without any additional hardware (apart from a USB cable).



3. Build and run 'Hello World'

3.1 Load the Project

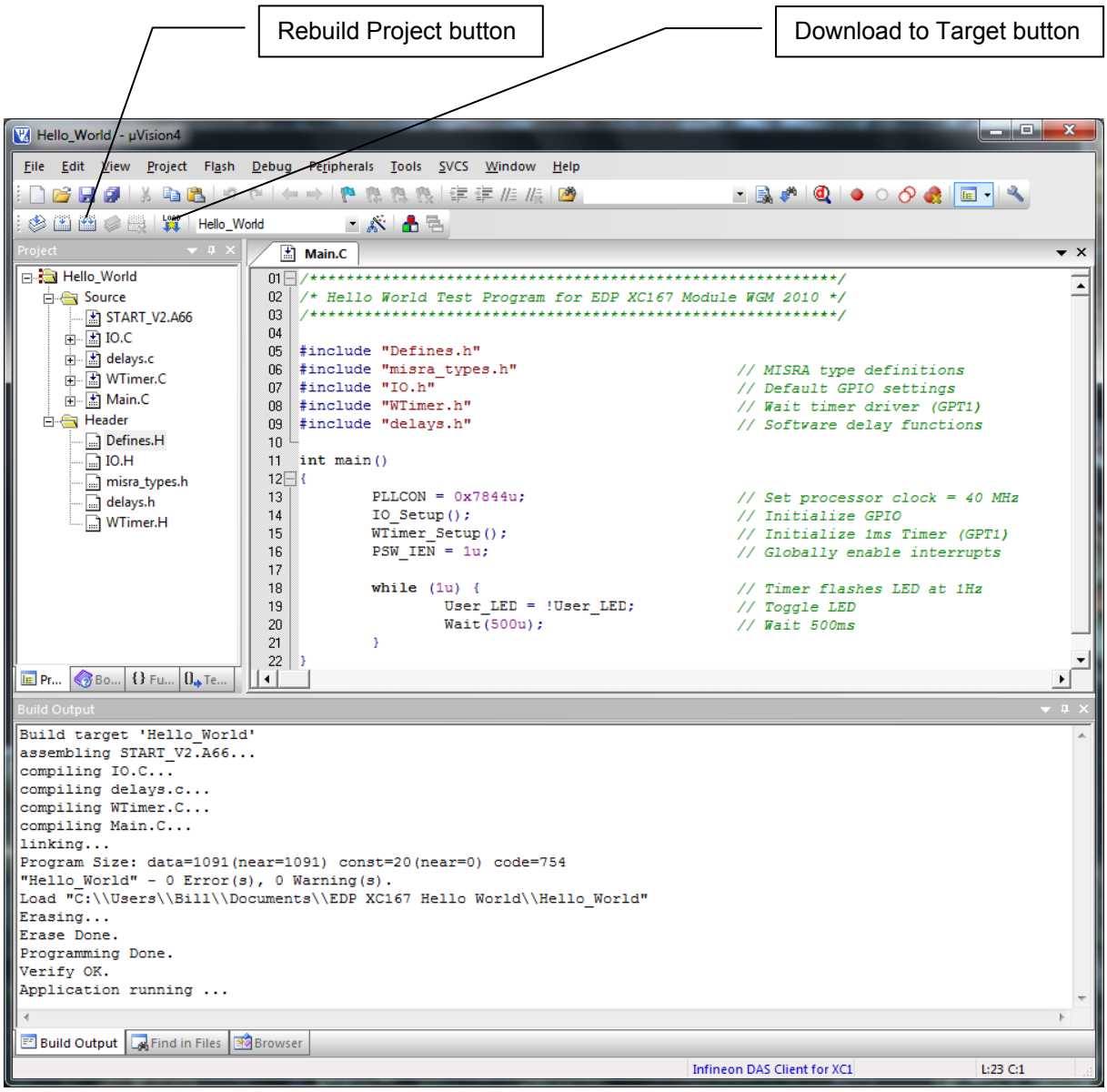
- Connect the Base Board to a USB port on the host computer using the debug connector.
- Turn on power to the Base Board.
- Download the 'EDP XC167 Hello World' folder from the EDP web site to your C: drive.

- Run Keil µVision 4 on the host computer.
- Click on **Open Project** from the Project menu and navigate to the 'EDP XC167 Hello World' folder. Double click on 'Hello_World' to load the project files.

Note that the folder already contains all the necessary source code files together with the Build files produced by the IDE. To demonstrate the procedure we will now perform the Build and Link operations.

3.2 Build the Project

- Click the **Rebuild** button on the Build toolbar (see picture below). The various files are compiled and linked, progress being reported in the Build Output pane.
- If there are any compilation errors they will be reported here and linking will not take place. Double clicking on a particular error message will take you to the offending line of source code in the edit pane above.



3.3 Download and program target Flash memory

- If zero errors are reported then the code can be downloaded to the target MCU and programmed into its Flash memory. Click on the **Download** button on the Build toolbar.
- should begin flashing indicating a successful download and program.

4. Build your own Project

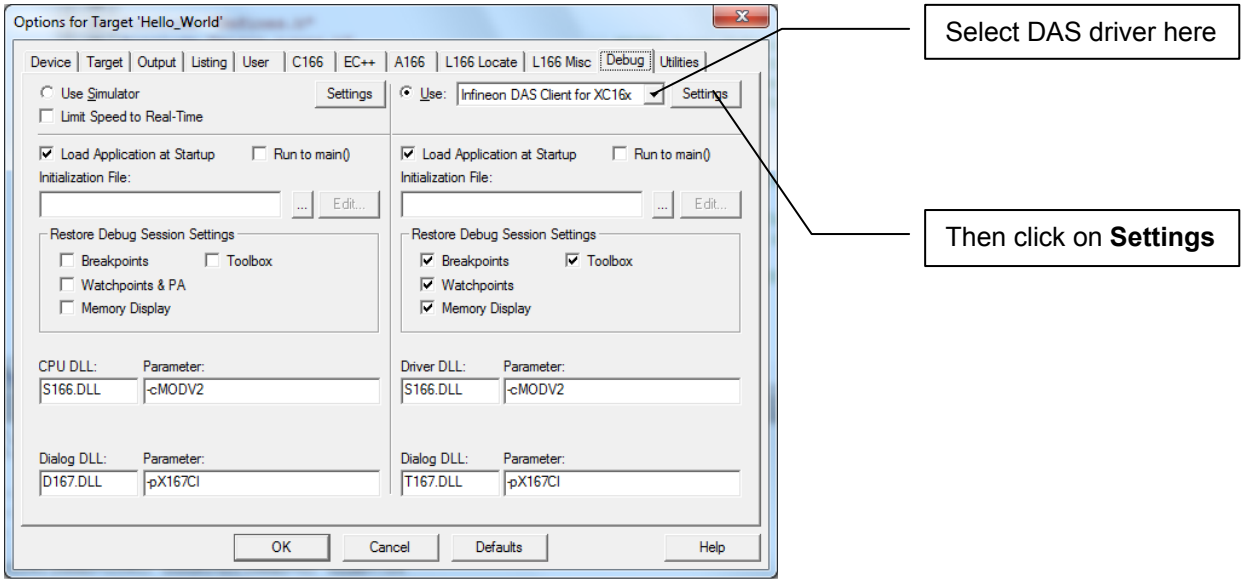
The procedure for building and programming your own project is exactly as outlined above using your own project folder containing source (.c) and header (.h) files.

The Hello World program does not use any EDP Application Module hardware but your projects probably will, so the necessary drivers will need to be included in your folder. Download these from the EDP web site. Note that MCU-specific drivers will also be required: see example above.

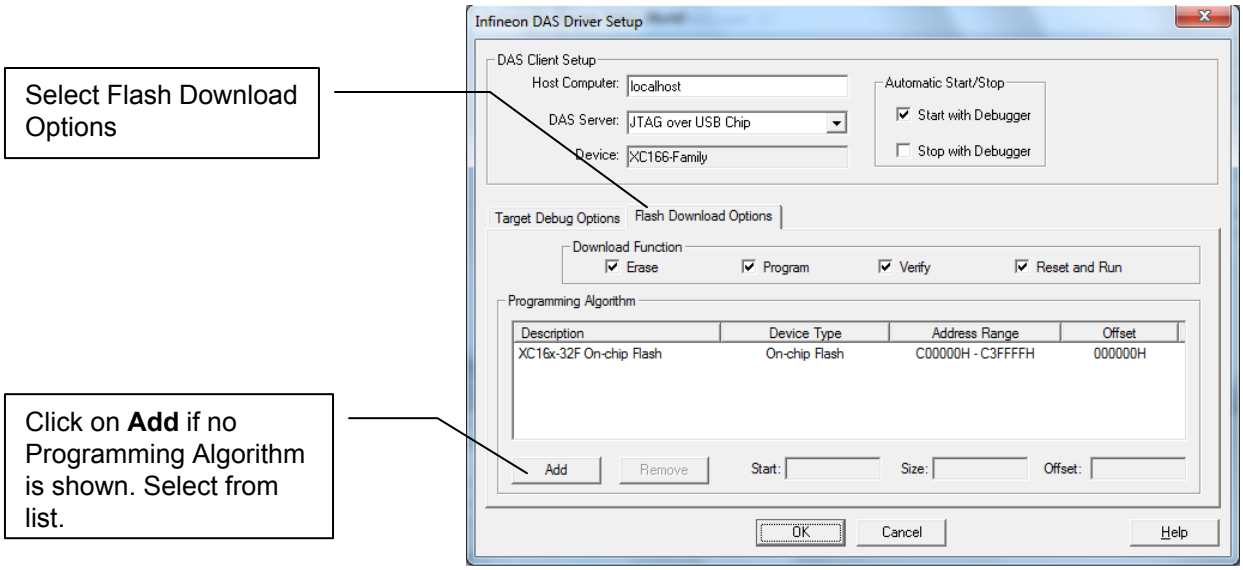
4.1 Setting up the Keil µVision IDE and DAS Client Driver

The IDE default for debug/programming driver must be switched to **Infineon DAS Client for XC16x**, when creating a new project. This is only required when Creating not Opening an existing project.

- Click on **New µVision Project** from the Project menu and navigate to a project folder where you keep your source files.
- Type in a project name and click on **Save**. You will now be asked to select a target MCU device. In this case select 'Infineon' and then the part number from the drop-down list.
- Right-click on Target and select Options for Target. **Select Debug** tab…



- Clicking on **Settings** takes you to Infineon DAS Driver Setup….



4.2 Creating a new Project

- Right-click on 'Source Group 1' in the Project pane (expanded from 'Target 1') and then on '**Add Files to Group**'. Select and **Add** all your source (.c) files. You can rename 'Target 1' and 'Source Group 1' and create other groups for other files by right-clicking on 'Target 1' and then clicking on **Manage Components…**
- Double-click on file: START_V2.A66 and select the **Configuration Wizard** tab. Look in the Peripheral Configuration settings menu. Make sure any peripherals you may be using are not disabled. These settings allow you to save power by disabling unused parts of the system. Resave the file when finished. There may be other settings that need changing, but the defaults should get things running.
- Build and load the project as before.

5. Conclusion

This is only a basic introduction to the Keil IDE; more complete information and a description of all the debugging features can be found on the Keil website.

Refer to the User Manual for these CM modules for details of alternative program development tools which can be used with EDP.

6. Appendix 1 EDP I²C Bus Device Addresses

8-bit address format is used: the LSB is reserved as the read-write bit. These addresses are defined in file: I2C-Directory.h

BaseBoard

BB_DIP = 0x40

8-bit DIP Switch

Address set by jumper link JP501 providing possible range of addresses: 0x40 (default), 0x42.

BB_EEPROM = 0xA2

Serial 4KB EEPROM memory

Address set by jumper links J601 to J603 providing possible range of addresses: 0xA0, 0xA2 (default), 0xA4, 0xA6, 0xA8, 0xAA, 0xAC, 0xAE.

Communications Module EDP-AM-CO1

CO1_RTC = 0xA0

Real-Time Clock and SRAM

Address set by jumper link J304 providing possible range of addresses: 0xA0 (default), 0xA2.

Digital I/O Module EDP-AM-DIO54

DIO54_Out = 0x46

Digital output latch

Address set by jumper links B305 to B307 providing possible range of addresses: 0x40, 0x42, 0x44, 0x46 (default), 0x48, 0x4A, 0x4C, 0x4E.

DIO54_In = 0x44

Digital input latch

Address set by jumper links B302 to B304 providing possible range of addresses: 0x40, 0x42, 0x44 (default), 0x46, 0x48, 0x4A, 0x4C, 0x4E.

Analogue Input Module EDP-AM-AN16

AN16_ADC = 0x6A

12-channel 10-bit Analogue to Digital Converter

AN16_Pot = 0x58

Digital Potentiometer for setting filter cut-off frequency of channels AN0 and AN1

Address set by jumper links J305 & J306 providing possible range of addresses: 0x58 (default), 0x5A, 0x5C, 0x5E.

Note that jumper links J204 & J205 select between CTRL_I2C (default) and I2C_GEN0 bus channels.

Dual BLDC Motor Module EDP-AM-MC2

MICROCHIP_MOTOR_DRIVER_BASE = 0x80

dsPIC U201

Address set by jumper links J201 to J203 providing possible range of addresses: 0x80 (default), 0x82, 0x84, 0x86, 0x88, 0x8A, 0x8C, 0x8E.

dsPIC U202

Address set by jumper links J205 to J207 providing possible range of addresses: 0x80, 0x82 (default), 0x84, 0x86, 0x88, 0x8A, 0x8C, 0x8E.

Note that spare addresses allow up to four MC2 modules per base board

Miscellaneous

SRF08 = 0xE0

Daventech SRF08 Ultrasonic Rangefinder module