Anybus® CompactCom Option Board STM

INSTALLATION GUIDE

HMSI-27-343 1.2 ENGLISH

Connecting Devices™

Important User Information

Liability

Every care has been taken in the preparation of this document. Please inform HMS Industrial Networks AB of any inaccuracies or omissions. The data and illustrations found in this document are not binding. We, HMS Industrial Networks AB, reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB. HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

There are many applications of this product. Those responsible for the use of this device must ensure that all the necessary steps have been taken to verify that the applications meet all performance and safety requirements including any applicable laws, regulations, codes, and standards.

HMS Industrial Networks AB will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features, timing, or functional side effects found outside the documented scope of this product. The effects caused by any direct or indirect use of such aspects of the product are undefined, and may include e.g. compatibility issues and stability issues.

The examples and illustrations in this document are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular implementation, HMS Industrial Networks AB cannot assume responsibility for actual use based on these examples and illustrations.

Intellectual Property Rights

HMS Industrial Networks AB has intellectual property rights relating to technology embodied in the product described in this document. These intellectual property rights may include patents and pending patent applications in the USA and other countries.

Table of Contents

Page

This page intentionally left blank

1 About This Document

1.1 About This Document

This manual describes how to install and configure the Anybus CompactCom 40 Option Board for the STM32 evaluation platform STM3240-EVAL from ST. The document includes software configuration guides for both the Keil µVision environment and IAR Embedded Workbench.

For additional related documentation and file downloads, please visit the support website at www.anybus.com/support.

1.2 Related Documents

Document	Author	Document ID
Anybus CompactCom 40 Software Design Guide	HMS	HMSI-216-125
Anybus CompactCom 40 Hardware Design Guide	HMS	HMSI-216-126

1.3 Document history

Version	Date	Description
1.0	2015-12-18	First revision
1.1	2016-02-15	Revised document
1.2	2017-03-13	Converted to DOX. IAR addition.

1.4 Trademark Information

Anybus® is a registered trademark of HMS Industrial Networks AB.

All other trademarks are the property of their respective holders.

1.5 Conventions

Ordered lists are used for instructions that must be carried out in sequence:

1. First do this
2. Then do this

Unordered (bulleted) lists are used for:

- Itemized information
- Instructions that can be carried out in any order

...and for action-result type instructions:

► This action...

  ➡ leads to this result

Bold typeface indicates interactive parts such as connectors and switches on the hardware, or menus and buttons in a graphical user interface.

```
Monospaced text is used to indicate program code and other
kinds of data input/output such as configuration scripts.
```

This is a cross-reference within this document: *Conventions, p. 4*

This is an external link (URL): www.hms-networks.com

ⓘ *This is additional information which may facilitate installation and/or operation.*

! This instruction must be followed to avoid a risk of reduced functionality and/or damage to the equipment, or to avoid a network security risk.

⚠ **Caution**
This instruction must be followed to avoid a risk of personal injury.

⚠ **WARNING**
This instruction must be followed to avoid a risk of death or serious injury.

2 Getting Started

2.1 General Information

This installation guide documents how to get the Anybus CompactCom device up and running, using the host application example code on the STM3240G_EVAL board.

Included is a basic description of what peripherals/features are used in the demo and how each one of them behaves. Also included is a detailed description of the hardware configuration that must be used to run the host application example code out-of-the-box.

2.2 Starter Kit Contents

- Anybus CompactCom 40 Option Board designed for the STM3240G-EVAL board

2.3 Other Required Items

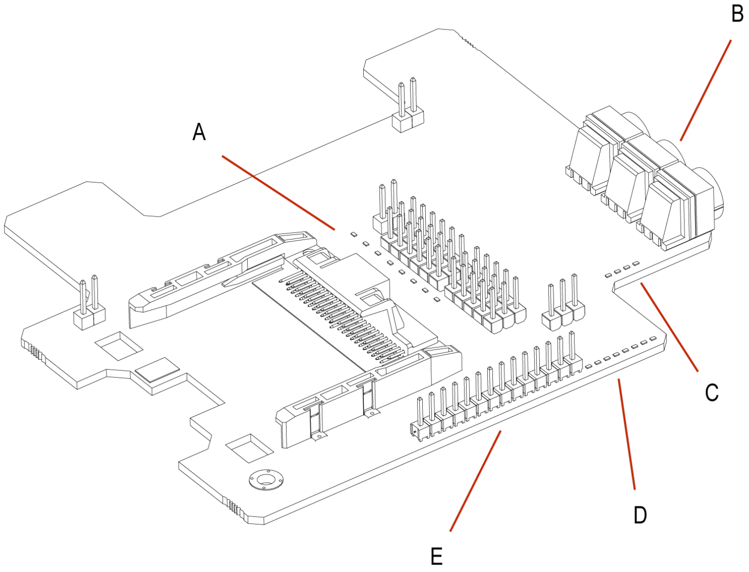
The following items are necessary, but not included in the starter kit:

- Anybus CompactCom module
- Zip file, available from the HMS website, containing the host application example code
- STM3240G-EVAL evaluation board
- Network cables
- Keil µVision or IAR Embedded Workbench

  Note: if the demo version of Keil µVision is used, see *Instructions for Keil µVision Demo Version, p. 17* for additional information.

2.4 Hardware Overview

2.4.1 Peripherals on the Option Board



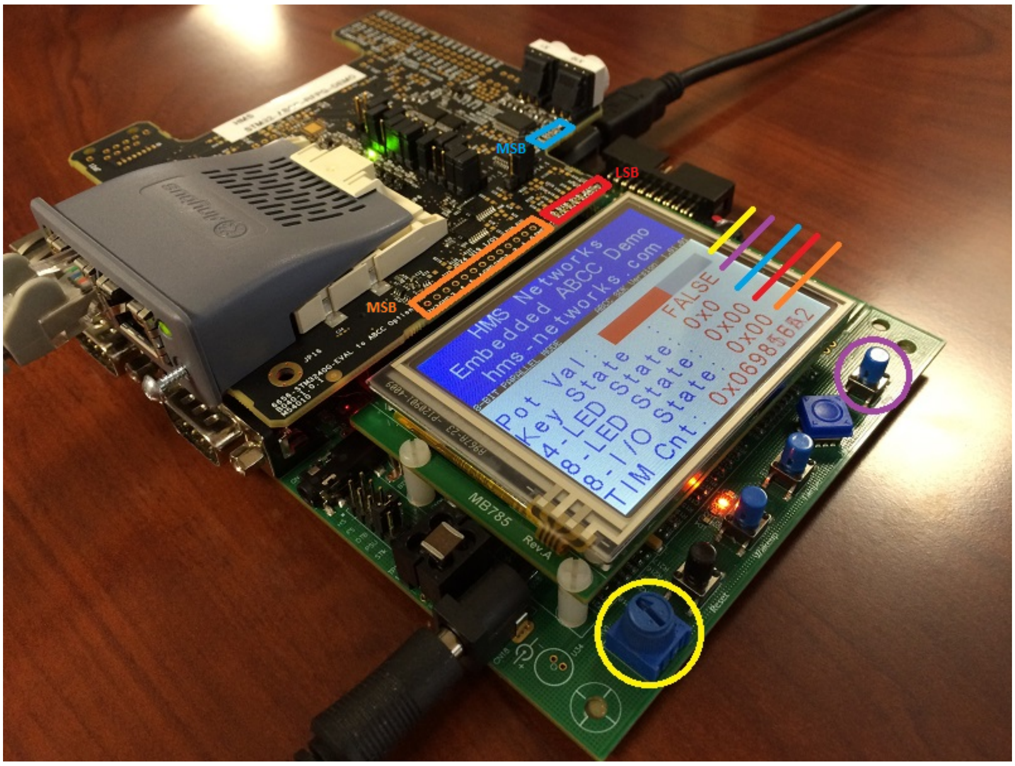
2.4.2 Peripherals Information

Item	Name	Description
A	8 LEDs array	H_LED1A - H_LED4B. Assignable via CompactCom. See the Anybus CompactCom Hardware Design Guide and Anybus CompactCom Software Design Guide for more information.
B	3 Hex switches	These switches are read via access to the 16-bit I2C IO expander. The switches are read via address 0x40. Bit 0-3: SW1 Bit 4-7: SW2 Bit 8-11: SW3
C	4 LEDs array	These LEDs are written to via access to the 16-bit I2C IO expander. They are available via address 0x40 (bits 12-15).
D	8 LEDs array	These LEDs are written to via access to the 16-bit I2C IO expander. They are available via address 0x44 (bits 0-7).
E	8 I/Os	These I/Os are read/written to via the 16-bit I2C expander. They are available via address 0x44 (bits 8-15).

For predefined help functions and information, see the file “appl_adimap_board_io.c” in the example code.

2.5 Assembly

Mount the Anybus CompactCom option board on the STM3240G-EVAL board. The four additional headers, provided with the starter kit, should be used for extra clearance. The picture below highlights (color-coded) the correlation between the values shown on the LCD and the hardware peripherals on the development board. The left-most pins/LEDs on the three bit-arrays (orange, red, blue) are the most significant bits of the values shown on the LCD.



2.6 Version Information

This demo was built and verified on the following software and hardware versions:

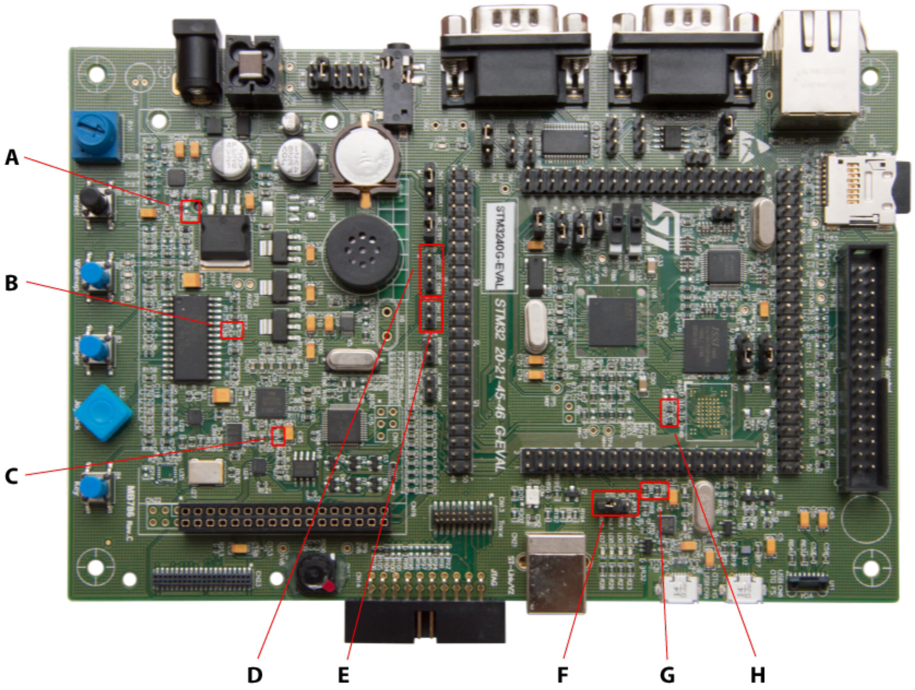
- Keil µVision version 4.74.0.22
- ARMCC version 5.03.0.76
- STM32CubeMX Version 4.9.0
- STM32Cube FW_F4 V1.7.0
- STM3240G-EVAL Rev C-04
- IAR Embedded Workbench for ARM 7.80.1.11873
- Host Application Example Code version 3.02.02

2.7 Hardware Configuration

The host application example code is set to run in parallel 8-bit mode, by default. To use this mode, follow the instructions as described in the *Parallel 8-bit/16-bit Interface, p. 9* section.

Additionally, ensure that the STM3240G-EVAL board’s JP18 is configured to “PSU” to utilize the power supply jack. This will guarantee power requirements of the option board and the Anybus CompactCom module are met.

The picture shows the STM3240G-EVAL board. The red framed parts, marked with position letters, indicate the locations of resistors and/or jumpers that are of importance when preparing the hardware for different operating modes. See specific operating modes below for details.



2.7.1 Parallel 8-bit/16-bit Interface

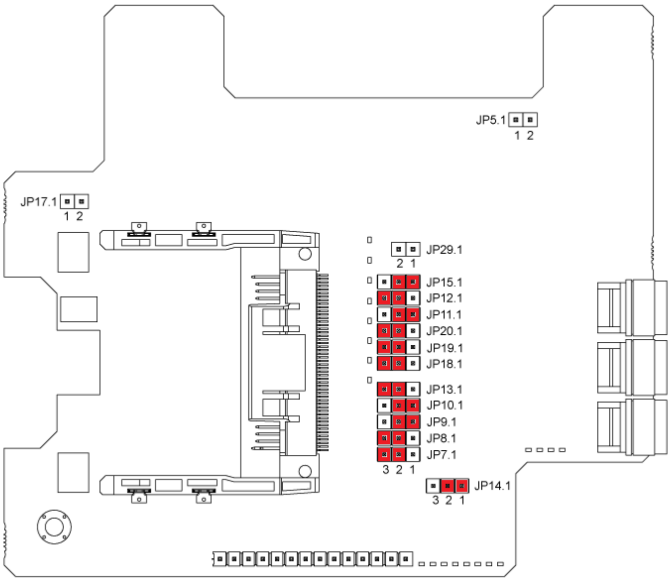
STM3240G-EVAL Hardware Configuration

The table below outlines all the necessary hardware changes (sorted by STM32 pin name) that must be performed for proper configuration, in order to use the Anybus CompactCom option board in parallel mode with the STM3240G-EVAL board.

STM32 Pin/Signal	STM3240G EVAL Card Position	Description
PF6/Smartcard_OFF	B	Remove R126 on EVAL-board to use PF6 as CompactCom Module Detect 0 signal (MD0).
PA4	C	Remove R115 on EVAL-board to use PA4 as CompactCom Module Identify 1 signal (MI1).
PF10	A	Remove R196 on EVAL-board to use PF10 as CompactCom Interrupt Request signal (H_IRQ_N).
PB15/OneNAND_INT	H	Remove R53 on EVAL-board to use PB15 as CompactCom RAM Loader signal.
PD6/FSMC_NWAIT	H	Remove R54 on EVAL-board to use PD6 as Protective Earth (PE) signal.
PI11/ULPI_DIR, PC0/ULPI_STP and PH4/ULPI_NXT	F, G	Unpopulate EVAL-board JP31 and remove R61 and R62.
PC9/MicroSDCard_D1/I2S_CKIN	D	Unpopulate EVAL-board JP16.

Anybus CompactCom Option Board Hardware Configuration for Parallel Mode

All jumpers in the table below are responsible for routing the STM32’s communication peripherals to the proper pins on the CompactCom connector. To set the communication interface to parallel, use the jumper settings here.



Operating Mode	Interface	Jumper	Setting
ABP_OP_MODE_8_BIT_PARALLEL ABP_OP_MODE_16_BIT_PARALLEL	FSMC #CS1 (8-bit) FSMC #CS1 (16-bit)	JP7	2-3
		JP8	2-3
		JP9	1-2
		JP10	1-2
		JP11	1-2
		JP12	2-3
		JP13	2-3
		JP14	1-2
		JP15	1-2
		JP18	2-3
		JP19	2-3
		JP20	2-3

2.7.2 SPI Interface

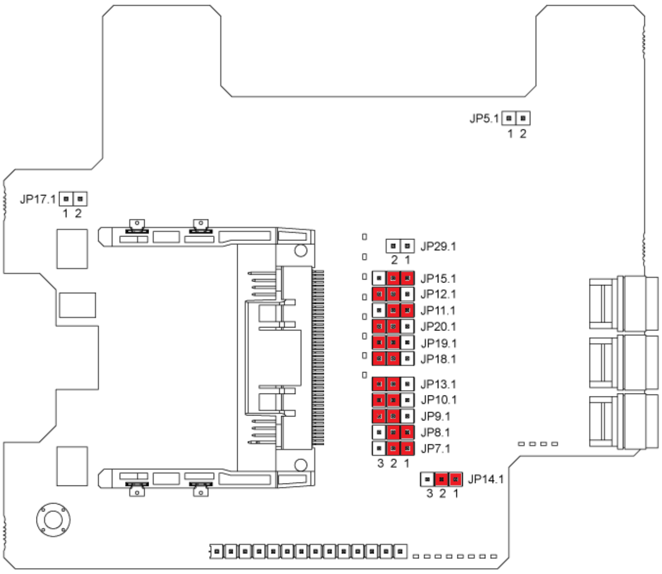
STM3240G-EVAL Hardware Configuration

The table below outlines all the necessary hardware changes (sorted by STM32 pin name) that must be performed for proper configuration, in order to use the Anybus CompactCom option board in SPI operation mode with the STM3240G-EVAL board.

STM32 Pin/Signal	STM3240G EVAL Card Position	Description
PF6/Smartcard_OFF	B	Remove R126 on EVAL-board to use PF6 as Compact-Com Module Detect 0 signal (MD0).
PA4	C	Remove R115 on EVAL-board to use PA4 as Compact-Com Module Identify 1 signal (MI1).
PF10	A	Remove R196 on EVAL-board to use PF10 as Compact-Com Interrupt Request signal (H_IRQ_N).
PB15/OneNAND_INT	H	Remove R53 on EVAL-board to use PB15 as Compact-Com RAM Loader signal.
PD6/FSMC_NWAIT	H	Remove R54 on EVAL-board to use PD6 as Protective Earth (PE) signal.
PI11/ULPI_DIR, PC0/ULPI_STP and PH4/ULPI_NXT	F, G	Unpopulate EVAL-board JP31 and remove R61 and R62.
PC9/MicroSDCard_D1/I2S_CKIN	D	Unpopulate EVAL-board JP16.

Anybus CompactCom Option Board Hardware Configuration for SPI Mode

All jumpers in the table below are responsible for routing the STM32's communication peripherals to the proper pins on the CompactCom connector. To set the communication interface to SPI, use the jumper settings here.



Operating Mode	Interface	Jumper	Setting
ABP_OP_MODE_SPI	SPI	JP7	1-2
		JP8	1-2
		JP9	2-3
		JP10	2-3
		JP11	1-2
		JP12	2-3
		JP13	2-3
		JP14	1-2
		JP15	1-2
		JP18	2-3
		JP19	2-3
		JP20	2-3

2.7.3 Serial Interface

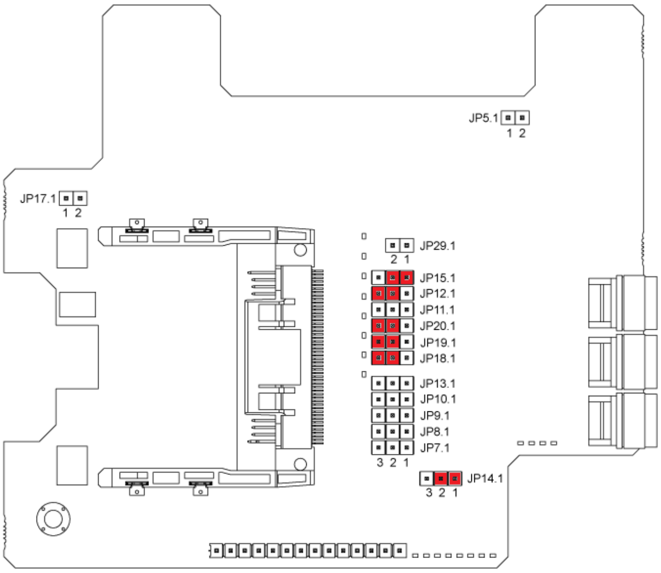
STM3240G-EVAL Hardware Configuration

The table below outlines all the necessary hardware changes (sorted by STM32 pin name) that must be performed for proper configuration, in order to use the Anybus CompactCom option board in serial mode with the STM3240G-EVAL board.

STM32 Pin/Signal	STM3240G EVAL Card Position	Description
PC6/I2S_MCK/ USART6_TX	E	Unpopulate EVAL-board JP21.
PF6/Smartcard_OFF	B	Remove R126 on EVAL-board to use PF6 as Compact-Com Module Detect 0 signal (MD0).
PA4	C	Remove R115 on EVAL-board to use PA4 as Compact-Com Module Identify 1 signal (MI1).
PF10	A	Remove R196 on EVAL-board to use PF10 as Compact-Com Interrupt Request signal (H_IRQ_N).
PB15/OneNAND_ INT	H	Remove R53 on EVAL-board to use PB15 as Compact-Com RAM Loader signal.
PD6/FSMC_NWAIT	H	Remove R54 on EVAL-board to use PD6 as Protective Earth (PE) signal.
PI11/ULPI_DIR, PC0/ULPI_STP and PH4/ULPI_NXT	F, G	Unpopulate EVAL-board JP31 and remove R61 and R62.
PC9/MicroSDCard_ D1/I2S_CKIN	D	Unpopulate EVAL-board JP16.

Anybus CompactCom Option Board Hardware Configuration for Serial Mode

All jumpers in the table below are responsible for routing the STM32's communication peripherals to the proper pins on the CompactCom connector. To set the communication interface to serial, use the jumper settings here.



Operating Mode	Interface	Jumper	Setting
ABP_OP_MODE_SERIAL_19_2 ABP_OP_MODE_SERIAL_57_6 ABP_OP_MODE_SERIAL_115_2 ABP_OP_MODE_SERIAL_625	UART6	JP7	X
		JP8	X
		JP9	X
		JP10	X
		JP11	X If safety is enabled in the Functional Safety Object (F8), it is recommended to set JP11 = 2-3.
		JP12	2-3
		JP13	X If safety is enabled in the Functional Safety Object (F8), set JP13 = 2-1. **IMPORTANT**: Other settings may damage the CompactCom or the Anybus CompactCom option board.
		JP14	1-2
		JP15	1-2
		JP18	2-3
		JP19	2-3
		JP20	2-3

'X' = Do not care

2.8 Demo Application

IMPORTANT: Before proceeding to mount the board and run the demo, assure that all steps in the Hardware Configuration section above have been covered. See *Hardware Configuration, p. 8*.

Failure to properly configure the hardware could result in damage to either the option card or the evaluation board!

For information about how to change or add to the code, see the Anybus CompactCom Host Application Implementation Guide, which can be downloaded from www.anybus.com.

2.8.1 Application Data Mapping

The demo is using a board specific ADI mapping specified in example_app/appl_adimap_board_io.c. The ADI:s are mapped on process data according to the tables below. The process data sizes are 3 bytes of output data and 15 bytes of input data.

Input Process Data

Byte	ADI number	ADI name	Description
1-4	100	Read TIM5	The STM32's 32-bit TIM5 counter configured with a tick of approximately 10ms. The current count of TIM5 is read using the callback "appl_GetTim5Cnt", which updates the internal variable "appl_lTimer".
5-6	101	Read RD Counter	16-bit counter incremented when any of the "get" callbacks are called (appl_Get3HexSw, appl_GetKey, and appl_GetPot).
7-8	102	Read WR Counter	16-bit counter incremented when either of the "set" callbacks are called (appl_Set4Leds or appl_Set8Leds8IOs).
9-10	103	Read 3 Hex Switches	The 3 Hex switches on the option board are read using the callback "appl_Get3HexSw".
11	106	Read Key	The "Key" pushbutton on the evaluation board is connected to GPIO PG15 on the STM32. The state is read using the callback "appl_GetKey".
12-15	107	Read Potentiometer	The RV1 potentiometer on the evaluation board is connected to one of the STM32's ADC peripherals (ADC1 Channel 7). This analog input is read using the callback "appl_GetPot", which converts the 12-bit ADC input into a 32-bit floating point number and scales to the equivalent 3.3V voltage level seen at the ADC input.

Output Process Data

Byte	ADI number	ADI name	Description
1	104	Read/Write 4 LEDs	The lower nibble controls the state of the four LEDs (D9-D12) on the option board. The states are controlled by the "appl_Set4Leds" callback.
2-3	105	Read/Write 8 LEDs + 8 I/Os	The lower 8-bits controls the state of the 8 LEDs (D13-D20) on the option board. The upper 8-bits controls the state of the 8 I/Os on JP24 on the option board. The states are controlled by the "appl_Set8Leds8IOs" callback.

Host Board’s MB785 LCD Module

The host board’s MB785 LCD module displays at 320x240 resolution and interfaces through the FMSC interface on chip-select #3.

It displays basic information about the platform and the current state of a subset of ADI variables.

The LCD-related code can be completely disabled by removing the “__USE_LCD” pre-compiler flag from the project settings.

Host Board’s RS232 Interface (CN16)

The standard input/output (STDIO) is retargeted to UART3 so standard routines such as printf and scanf can be used on this interface. A basic boot-up message is printed to the standard output on system start-up along with the “ABCC_PORT_DebugPrint” output.

Serial settings: 115200-8-N-1.

2.8.2 Instructions for Keil µVision

To build and run the demo application, follow the steps below.

1. Open the project file “STM3240G_ABCC_Option_Card.uvproj” in Keil µVision.
   a. Project->Open project...
   b. Browse to and select the file name “STM3240G_ABCC_Option_Card.uvproj”
2. In “abcc_drv_cfg.h”, define the following macros according to the configured hardware communication interface in section 2.7.

```
#define ABCC_CFG_DRV_PARALLEL        ( TRUE )
#define ABCC_CFG_DRV_PARALLEL_30     ( FALSE )
#define ABCC_CFG_DRV_SPI             ( FALSE )
#define ABCC_CFG_DRV_SERIAL          ( FALSE )
```

3. In “abcc_drv_cfg.h”, define the operating mode to use.

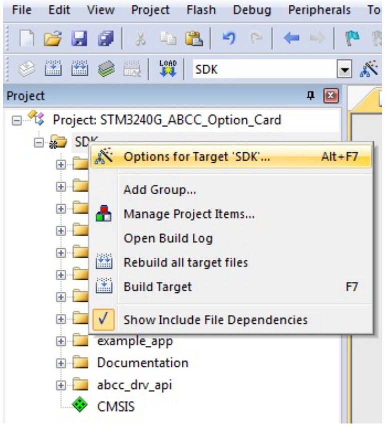
```
#define ABCC_CFG_ABCC_OP_MODE_40   ABP_OP_MODE_8_BIT_PARALLEL
#define ABCC_CFG_ABCC_OP_MODE_30   ABP_OP_MODE_8_BIT_PARALLEL
```

4. Build the project.
   a. Project->Build target (or press F7)
5. Flash the target software.
   a. With the STM3240G-EVAL board’s integrated programmer plugged into the PC (via the USB type-B port), select: Debug->Start/Stop debug session
6. Run the demo.
   a. Debug->Run (or press F5)
7. The demo should now be running. Configure the master/PLC on the network and start exchanging data.

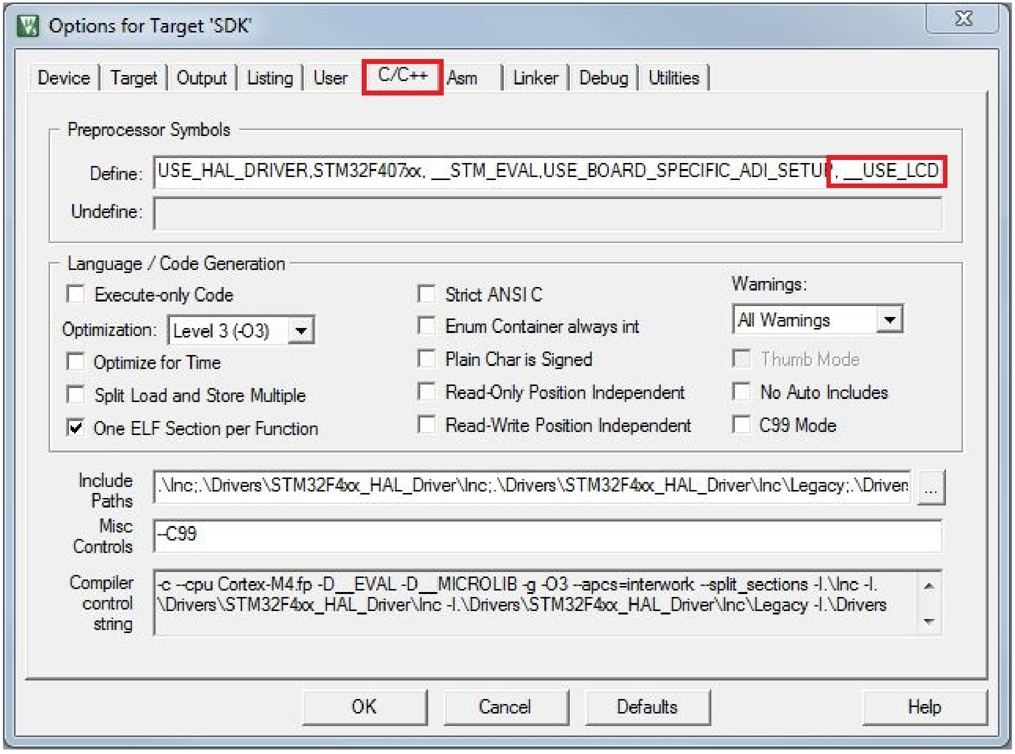
2.8.3 Instructions for Keil µVision Demo Version

Following the steps below will result in a build that is smaller than 32kB, which is the limit in the demo version of Keil µVision.

1. Right click the project folder (SDK) and choose "Option for Target 'SDK'..".



2. Click on the "C/C++" tab and remove ", __USE_LCD".



Preprocessor Symbols

Define: USE_HAL_DRIVER,STM32F407xx, __STM_EVAL,USE_BOARD_SPECIFIC_ADI_SETUP, __USE_LCD

Undefine:

Language / Code Generation

- □ Execute-only Code
- Optimization: Level 3 (-O3)
- □ Optimize for Time
- □ Split Load and Store Multiple
- ☑ One ELF Section per Function
- □ Strict ANSI C
- □ Enum Container always int
- □ Plain Char is Signed
- □ Read-Only Position Independent
- □ Read-Write Position Independent
- Warnings: All Warnings
- □ Thumb Mode
- □ No Auto Includes
- □ C99 Mode

Include Paths: .\Inc;.\Drivers\STM32F4xx_HAL_Driver\Inc;.\Drivers\STM32F4xx_HAL_Driver\Inc\Legacy;.\Driver

Misc Controls: --C99

Compiler control string:

```
-c --cpu Cortex-M4.fp -D__EVAL -D__MICROLIB -g -O3 --apcs=interwork --split_sections -I.\Inc -I.
\Drivers\STM32F4xx_HAL_Driver\Inc -I.\Drivers\STM32F4xx_HAL_Driver\Inc\Legacy -I.\Drivers
```

3. Open the file "\abcc_adapt\abcc_drv_cfg.h".

4. Set either "ABCC_CFG_DRV_PARALLEL_30" or "ABCC_CFG_DRV_SERIAL" to FALSE depending on which operating mode that is used.

5. Hit F7 to compile.

2.8.4 Instructions for IAR Embedded Workbench

Demo application for IAR Embedded Workbench for ARM project has two build configurations as listed below.

- EV-Debug
- 32KBLV-Debug

“32KBLV-Debug” configuration will build a demo executable with size <32KB without support of LCD display interface. “EV-Debug” configuration, on the other hand, will build a demo executable with all features including LCD display interface.

The standard input/output (STDIO) is retargeted to Terminal I/O Window of IAR Embedded Workbench. Standard routines such as printf and scanf can be used on this interface. A basic boot-up message is printed to the standard output on system start-up.

IAR Embedded Workbench for ARM can be downloaded from URL below.

https://www.iar.com/iar-embedded-workbench/#!?currentTab=free-trials

After download and installation, you have the following evaluation options to choose from:

- 30-day time limited but fully functional license
- Size limited kickstart license without any time limit

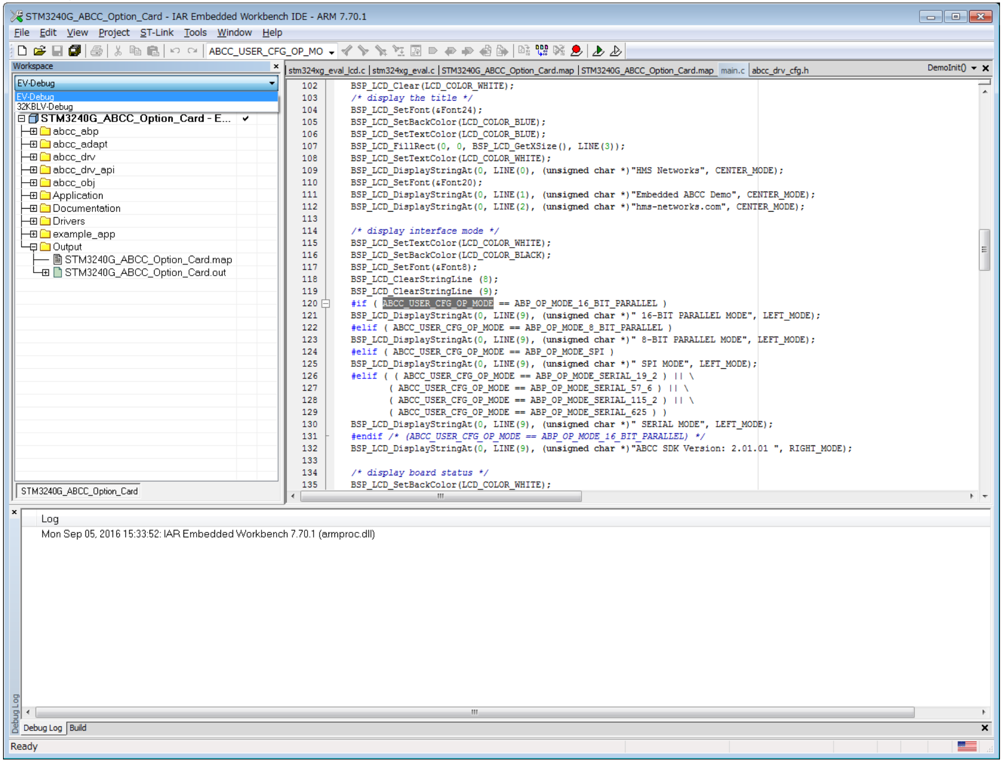
If you have the 30-day time limited evaluation license, then you may use both “EV-Debug” and “32KBLV-Debug” build configurations.

With the size limited kickstart license, please test with “32KBLV-Debug” configuration only.

Follow the steps below to build and execute the demo application with the STM32 evaluation board.

1. Startup IAR Embedded Workbench and Open "STM3240G_ABCC_Option_Card.eww" workspace.

2. Choose build configuration from project build configuration selection window. Select “EV-Debug” for IAR Embedded Workbench license without size limitation.



STM3240G_ABCC_Option_Card - IAR Embedded Workbench IDE - ARM 7.70.1
File Edit View Project ST-Link Tools Window Help
ABCC_USER_CFG_OP_MO
Workspace
EV-Debug
EV-Debug
32KBLV-Debug
STM3240G_ABCC_Option_Card - E...
abcc_abp
abcc_adapt
abcc_drv
abcc_drv_api
abcc_obj
Application
Documentation
Drivers
example_app
Output
STM3240G_ABCC_Option_Card.map
STM3240G_ABCC_Option_Card.out
STM3240G_ABCC_Option_Card
stm324xg_eval_lcd.c
stm324xg_eval.c
STM3240G_ABCC_Option_Card.map
STM3240G_ABCC_Option_Card.map
main.c
abcc_drv_cfg.h
DemoInit()
Log
Debug Log
Build
Ready

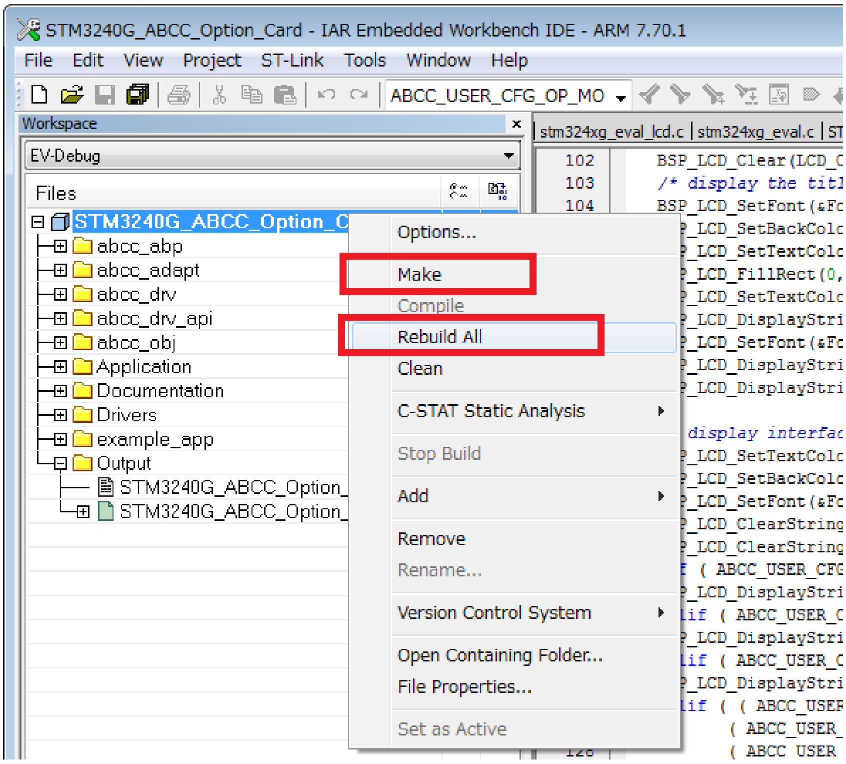
```
    BSP_LCD_Clear(LCD_COLOR_WHITE);
    /* display the title */
    BSP_LCD_SetFont(&Font24);
    BSP_LCD_SetBackColor(LCD_COLOR_BLUE);
    BSP_LCD_SetTextColor(LCD_COLOR_BLUE);
    BSP_LCD_FillRect(0, 0, BSP_LCD_GetXSize(), LINE(3));
    BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
    BSP_LCD_DisplayStringAt(0, LINE(0), (unsigned char *)"HMS Networks", CENTER_MODE);
    BSP_LCD_SetFont(&Font20);
    BSP_LCD_DisplayStringAt(0, LINE(1), (unsigned char *)"Embedded ABCC Demo", CENTER_MODE);
    BSP_LCD_DisplayStringAt(0, LINE(2), (unsigned char *)"hms-networks.com", CENTER_MODE);

    /* display interface mode */
    BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
    BSP_LCD_SetBackColor(LCD_COLOR_BLACK);
    BSP_LCD_SetFont(&Font8);
    BSP_LCD_ClearStringLine (8);
    BSP_LCD_ClearStringLine (9);
    #if ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_16_BIT_PARALLEL )
    BSP_LCD_DisplayStringAt(0, LINE(9), (unsigned char *)" 16-BIT PARALLEL MODE", LEFT_MODE);
    #elif ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_8_BIT_PARALLEL )
    BSP_LCD_DisplayStringAt(0, LINE(9), (unsigned char *)" 8-BIT PARALLEL MODE", LEFT_MODE);
    #elif ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_SPI )
    BSP_LCD_DisplayStringAt(0, LINE(9), (unsigned char *)" SPI MODE", LEFT_MODE);
    #elif ( ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_SERIAL_19_2 ) || \
            ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_SERIAL_57_6 ) || \
            ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_SERIAL_115_2 ) || \
            ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_SERIAL_625 ) )
    BSP_LCD_DisplayStringAt(0, LINE(9), (unsigned char *)" SERIAL MODE", LEFT_MODE);
    #endif /* (ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_16_BIT_PARALLEL) */
    BSP_LCD_DisplayStringAt(0, LINE(9), (unsigned char *)"ABCC SDK Version: 2.01.01 ", RIGHT_MODE);

    /* display board status */
    BSP_LCD_SetBackColor(LCD_COLOR_WHITE);
```

```
Mon Sep 05, 2016 15:33:52: IAR Embedded Workbench 7.70.1 (armproc.dll)
```

3. Open “abcc_drv_cfg.h” file in the IAR Embedded Workbench editor, and set up the following macros according to the configured hardware communication interface in section 2.7.

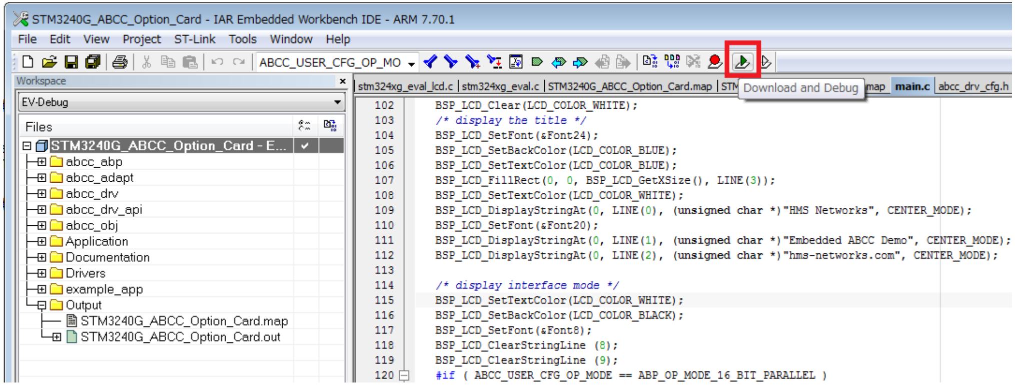
#define ABCC_CFG_DRV_PARALLEL	(TRUE)
#define ABCC_CFG_DRV_PARALLEL_30	(FALSE)
#define ABCC_CFG_DRV_SPI	(FALSE)
#define ABCC_CFG_DRV_SERIAL	(FALSE)

4. Right click the project title and run “Make” or “Rebuild All”. Else, you may simply push F7 to build the project for the selected configuration.



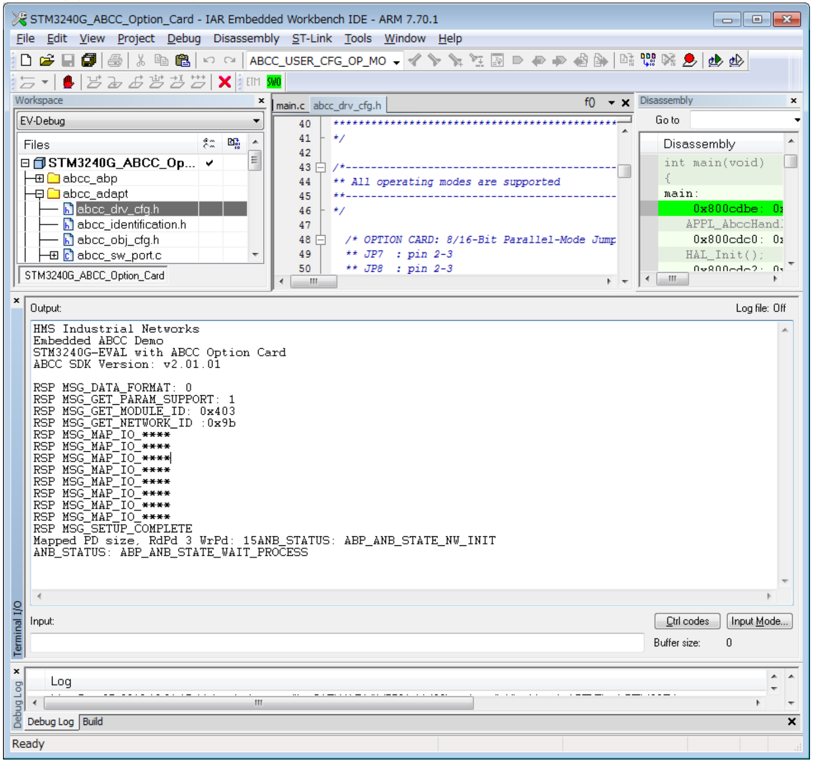
5. Connect on-board ST-Link debug port USB cable to PC. Power-On the evaluation board.

6. Push “Debug and Download” icon button from IAR Embedded Workbench command menu bar to start loading executable to STM32 flash. Alternatively, you may use “Ctrl+D” key to start debug session.


STM3240G_ABCC_Option_Card - IAR Embedded Workbench IDE - ARM 7.70.1
File Edit View Project ST-Link Tools Window Help
ABCC_USER_CFG_OP_MO
Workspace
EV-Debug
Files
STM3240G_ABCC_Option_Card - E...
abcc_abp
abcc_adapt
abcc_drv
abcc_drv_api
abcc_obj
Application
Documentation
Drivers
example_app
Output
STM3240G_ABCC_Option_Card.map
STM3240G_ABCC_Option_Card.out
stm324xg_eval_lcd.c
stm324xg_eval.c
STM3240G_ABCC_Option_Card.map
main.c
abcc_drv_cfg.h
Download and Debug

```
BSP_LCD_Clear(LCD_COLOR_WHITE);
/* display the title */
BSP_LCD_SetFont(&Font24);
BSP_LCD_SetBackColor(LCD_COLOR_BLUE);
BSP_LCD_SetTextColor(LCD_COLOR_BLUE);
BSP_LCD_FillRect(0, 0, BSP_LCD_GetXSize(), LINE(3));
BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
BSP_LCD_DisplayStringAt(0, LINE(0), (unsigned char *)"HMS Networks", CENTER_MODE);
BSP_LCD_SetFont(&Font20);
BSP_LCD_DisplayStringAt(0, LINE(1), (unsigned char *)"Embedded ABCC Demo", CENTER_MODE);
BSP_LCD_DisplayStringAt(0, LINE(2), (unsigned char *)"hms-networks.com", CENTER_MODE);

/* display interface mode */
BSP_LCD_SetTextColor(LCD_COLOR_WHITE);
BSP_LCD_SetBackColor(LCD_COLOR_BLACK);
BSP_LCD_SetFont(&Font8);
BSP_LCD_ClearStringLine (8);
BSP_LCD_ClearStringLine (9);
#if ( ABCC_USER_CFG_OP_MODE == ABP_OP_MODE_16_BIT_PARALLEL )
```

7. Open the Terminal I/O Window with “View -> Terminal I/O” menu bar command. Once the debugger is ready, press “F5” to Run the loaded executable.

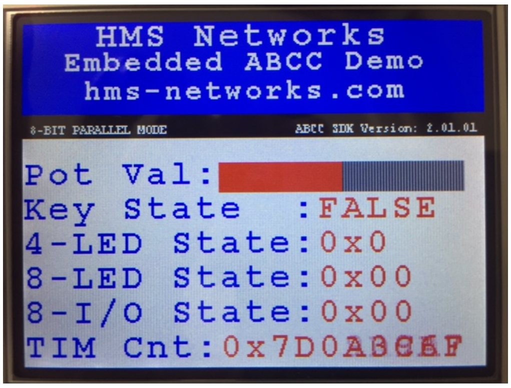


```
*****************************************************
*/

/*-----------------------------------------------------
** All operating modes are supported
**-----------------------------------------------------
*/

  /* OPTION CARD: 8/16-Bit Parallel-Mode Jump
  ** JP7  : pin 2-3
  ** JP8  : pin 2-3
```

```
int main(void)
{
main:
    0x800cdbe:  0:
  APPL_AbccHand:
    0x800cdc0:  0:
  HAL_Init();
    0x800cdc2:  0:
```

```
HMS Industrial Networks
Embedded ABCC Demo
STM3240G-EVAL with ABCC Option Card
ABCC SDK Version: v2.01.01

RSP MSG_DATA_FORMAT: 0
RSP MSG_GET_PARAM_SUPPORT: 1
RSP MSG_GET_MODULE_ID: 0x403
RSP MSG_GET_NETWORK_ID :0x9b
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_MAP_IO_****
RSP MSG_SETUP_COMPLETE
Mapped PD size, RdPd 3 WrPd: 15ANB_STATUS: ABP_ANB_STATE_NW_INIT
ANB_STATUS: ABP_ANB_STATE_WAIT_PROCESS
```

8. The demo should now be running. Configure the master/PLC on the network and start exchanging data. If you are running with “EV-Debug” build configuration, the LCD display on the evaluation board will show various status outputs.



Box 4126
300 04 Halmstad, Sweden

info@hms.se