UM0935
User manual

STEVAL-PCC009V2, IBU universal interface based on the STM32F103RB

Introduction

The IBU universal interface (UI) is a tool which contains:

- An IBU UI board (STEVAL-PCC009V2), which is an STM32-based USB-to-serial interface bridge consisting of a configurable 10-pin and 30-pin interface
- Dll files which are available software resources that allow the user to develop customized GUIs as per application requirements.

This user manual explains the functions of the IBU UI tool (STEVAL-PCC009V2) and how to use it. IBU UI is a complete tool to rapidly develop application prototypes. On this demonstration board, the STM32 microcontroller is used as the interface between the PC and the end device. Due to intelligence available in the STM32 device, various communication peripherals are multiplexed with GPIOs and ADC and PWM channels in both a 10-pin and 30-pin interface.

In these interfaces, there is a provision to connect a device which can communicate using I^2C, SPI and UART. Therefore, the IBU UI tool allows the user to connect a serial communication based device to the PC. At the same time it allows the user to control some GPIOs available in 10-pin and 30-pin interfaces and set them in input/output modes, as per application requirements.

Power to the board is provided from a USB mini B-type connector.

DLL files are provided with this tool so that the user can make their own customized PC GUI as per requirements.

Therefore the IBU UI tool, by taking care of all the microcontroller complexities, provides an option for the end user to focus on its application development, therefore increasing its efficiency and time to market.

The IBU UI tool supports two modes:

- Application mode: this PC GUI allows interfacing of the SPI, I^2C and UART interface and controlling the communication parameters with the help of the GUI itself
- DFU mode: this mode allows the user to change the firmware, if required, to suit its applications.

Contents

List of tables

List of figures

1 Section organization of the user manual

The user must go through *Section 2.1* to *Section 2.4* of this manual to perform the initial setup that is required to run the IBU UI tool. After reading these sections, the user can understand how to install the software and hardware setup.

To choose how to select a communication interface, the user must read *Section 2.5*, this section explains the seven modes (as shown in *Table 1*) that are available in the 10-pin and 30-pin header and how to select which interface mode is best suited to the development of their application.

After this, the user must decide which communication interface of a 10-pin or 30-pin header should be used. As shown in *Table 1* below, *Section 3* is documented in such a manner that the user need only refer to the section corresponding to the communication interface to be used. For instance, if the user wants to use the UART2 interface mode of the 30-pin header, they need only refer to *Section 3.7* of this document.

Table 1. Section to be referred to for a particular mode of a 10-pin or 30-pin interface

Header	Interface	Refer to section
10-pin header	I^2C mode	*3.1*
	SPI mode	*3.2*
	UART mode	*3.7*
30-pin header	I^2C mode	*3.7*
	SPI mode	*3.7*
	UART1 mode	*3.7*
	UART2 mode	*3.7*

Section 4 explains how to use the DFU capability of this tool. This section needs to be referred to when there it is necessary to update the firmware of the microcontroller of this tool.

2 Getting started

2.1 System requirements

In order to use the IBU universal interface (IBU UI) tool with a Windows operating system, a recent version of Windows, such as Windows 2000 or Windows XP, must be installed on the PC.

The version of the Windows OS installed on the PC may be determined by clicking on the “System” icon in the control panel.

2.2 Package contents

The IBU UI tool includes the following items:

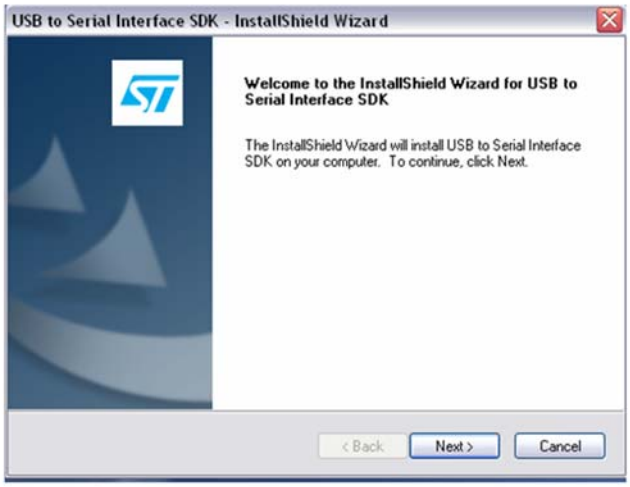
- Hardware content:
  - One board
  - BOM list
  - Schematic
- Software content:
  - DFU firmware
  - DLL files of the I^2C, SPI and UART interface of the 10-pin header
  - DLL files of the I^2C, SPI, and UART1 and UART2 interface of the 30-pin header
  - Source code (including DFU)
- Documentation:
  - User manual (to work in functional mode)
  - User manual (to work in DFU mode)
  - Help file on how to use the DLL file

2.3 Software installation

The DLLs are provided with the tool, mainly in the form of a CD as a part of the package. The folder contains the setup files.

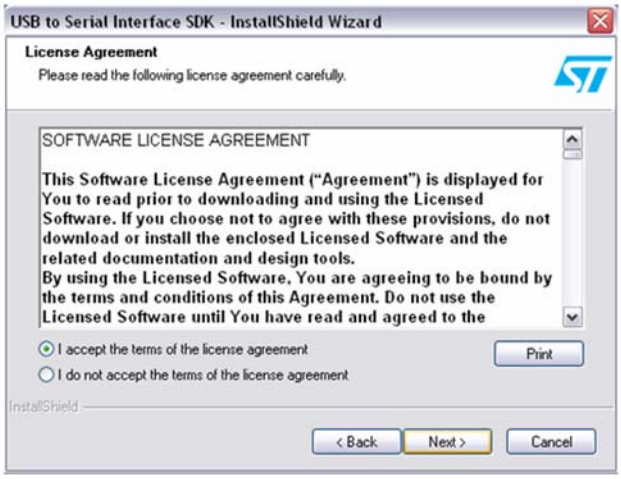
- Step1: as soon as the user clicks the setup.exe icon, the following window appears:

Figure 1. Installation Window



- Step 2: read the license file and click the “Next” button if you accept the license.

Figure 2. License Window



- Step 3: select the folder in which to install the software. By default it installs the software in the following path: C:\Program Files\STMicroelectronics\USB to serial interface SDK\Dll&Libraries

Figure 3. Destination folder

- Step 4: after selecting the folder and clicking the “Next” button, the software starts installing.

Figure 4. Installation ongoing

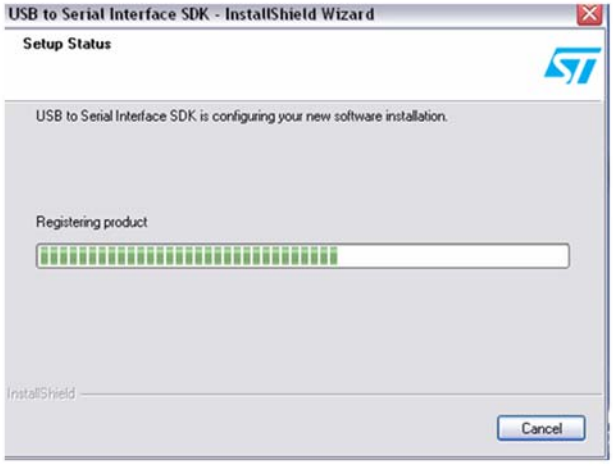
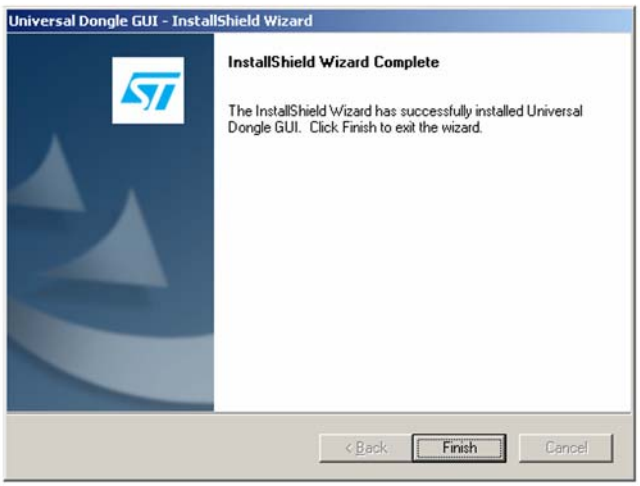


Figure 5. Installation complete

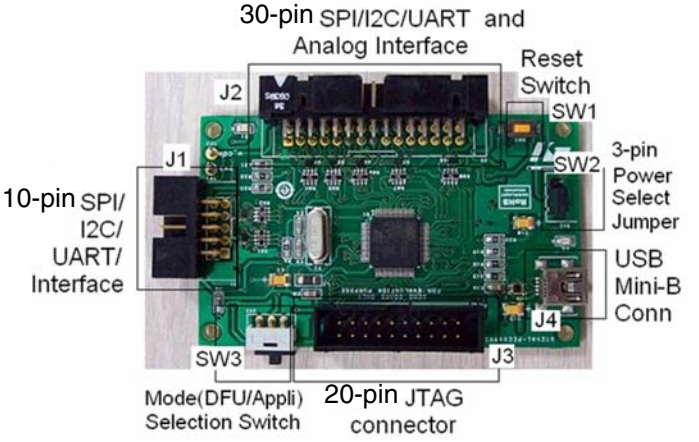


After clicking the “Finish” button, the software is installed in the directory selected or in the default directory. The shortcut of this software is also available in the Start menu. The help file on how to use DLL is also available in the same directory.

2.4 Hardware installation

Figure 6 below shows a snapshot of the IBU UI board.

Figure 6. STEVAL-PCC009V2, IBU universal interface board



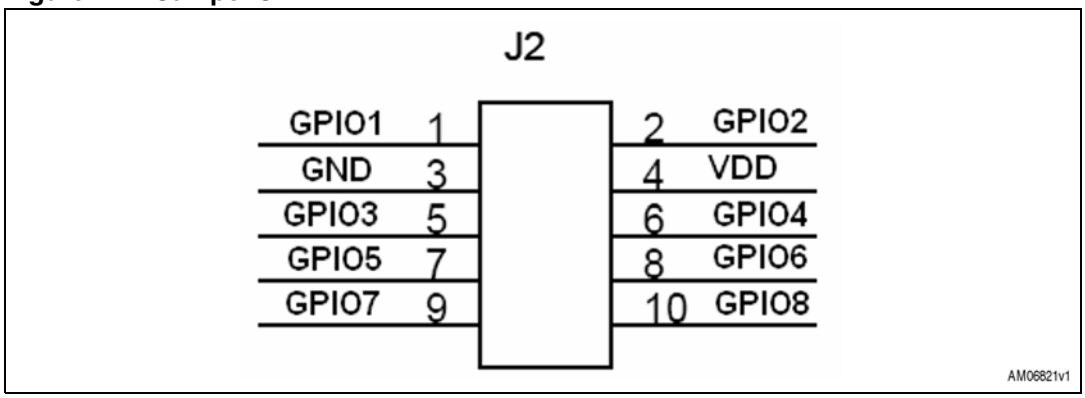
2.4.1 Power supply

The board is directly powered by the USB mini B-type connector J4 (bus powered). There is a power LED D2 available onboard, as soon as the board is powered, using the USB mini-B cable, it lights up.

2.4.2 Jumper/header settings

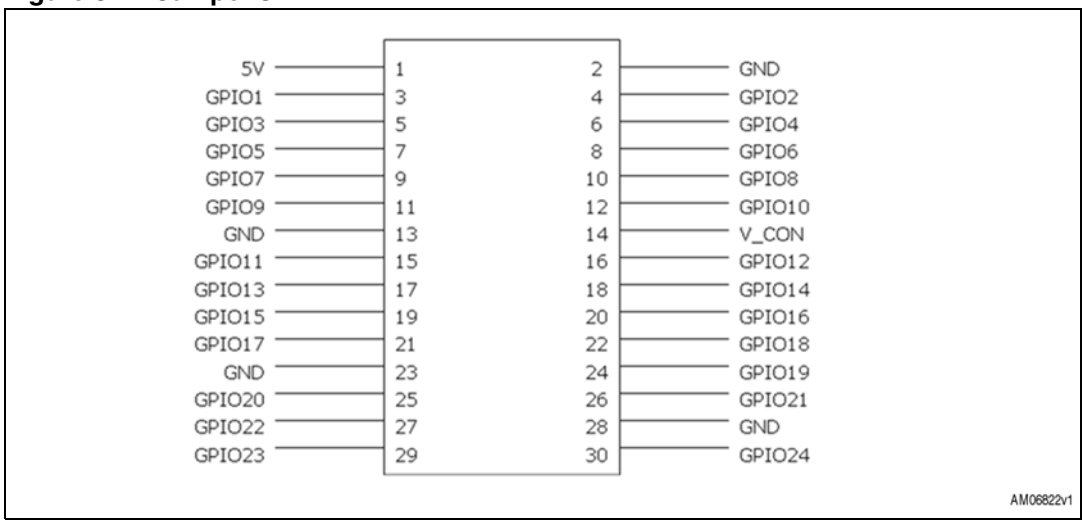
- J1: J1 is the 10-pin header available on the IBU UI board. There are 8 GPIOs, GND, and VDD (3.3 V) lines available, as shown in *Figure 7* below:

Figure 7. Jumper J1



- J2: J2 is the 30-pin header available on the IBU UI board. There are 24 GPIOs, GND, and VDD lines available, as shown in *Figure 8* below:

Figure 8. Jumper J2



- J3: This is the standard 20-pin JTAG header available on the board. This can be used by the user to run the board in debug mode using any JTAG based debugger for an STM32 device.
- SW1: this is the reset switch that can be used to reset the board at any point.

2.5 Selection of the interface

The tool has 10-pin and 30-pin interface headers. Both of these headers support various communication peripherals, as shown in *Table 2*.

These headers and their corresponding pins can be used in various modes and GPIO configurations.

Table 2. Availability of various communication peripherals and GPIOs on 10-pin and 30-pin interfaces

Interfaces	30-pin interface	10-pin interface
I^2C	1	1
SPI	1	1
UART(SCI)	2	1
PWM GPIOs	4	2
ADC channels	4	1

As shown in *Table 3*, the user can configure the IBU UI tool in 7 modes, 3 of these modes are on a 10-pin header and 4 on a 30-pin header. For instance, if the user mainly aims at using an I^2C communication interface, there are two available choices:

a) I^2C mode of a 10-pin header: along with the communication peripheral I^2C, the user has 6 GPIOs, of which 2 GPIOs can be used as PWM channels and 1 can be used as an ADC channel

b) I^2C mode of 30-pin header: along with the communication peripheral I^2C, the user has 22 GPIOs, of which 4 GPIOs can be used as PWM channels and 4 can be used as ADC channels

Table 3. Number of total GPIOs, PWM GPIOs, and ADC channels in 10-pin and 30-pin headers in various modes

Header	Interfaces modes	Total GPIOs	PWM GPIOs	ADC channels
10-pin header	I^2C mode	6	2	1
	SPI mode	4	2	1
	UART mode	4	2	1
30-pin header	I^2C mode	22	4	4
	SPI mode	20	4	4
	UART1 mode	20	4	4
	UART2 mode	22	4	4

Please refer to *Appendix B*, *Table 5* to understand the possible GPIO modes and communication interfaces available on each pin in 10-pin headers. Refer also to *Appendix C*, *Table 6* to understand the possible GPIO modes and communication interfaces available on each pin in 30-pin headers.

Based on the above description, the user is able to select which mode is most suited to their application development.

Please note that any two communication interfaces of 10-pin or 30-pin headers cannot be used at the same time. For instance, the user cannot use the I^2C mode of 10-pin headers and the I^2C mode of 30-pin headers at the same time or use the I^2C and SPI mode at the same time. To switch between the 7 modes available the user needs to select, using DLL, the interface to be used. As the user switches between the two modes, the settings of the previous mode are reset. For Instance, if the user is using the I^2C mode of a 10-pin header and switches to the I^2C mode of a 30-pin header, the settings of the previous 10-pin header are reset and all the pins of the 10-pin header go into input pull-up mode.

3 Running the IBU UI tool

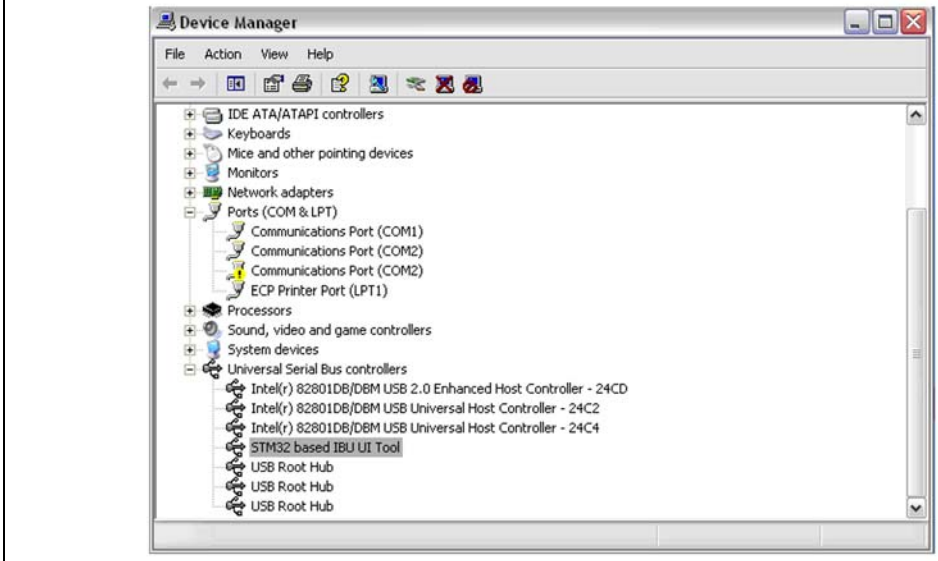
To run the board, connect it to the PC with the USB mini B-type cable.

As soon as the board is powered using the USB mini-B cable, power LED D2 lights up. If this LED fails to light up, take the following steps:

1. Check if the USB cable is working properly or not
2. Press the SW1 reset button.

As a result, the board should be enumerated as an IBU universal interface tool and it is shown as “STM32 based IBU UI Tool”, as shown in Figure 9, in the device manager window. If this message does not appear, please contact technical support.

Figure 9. Enumeration result



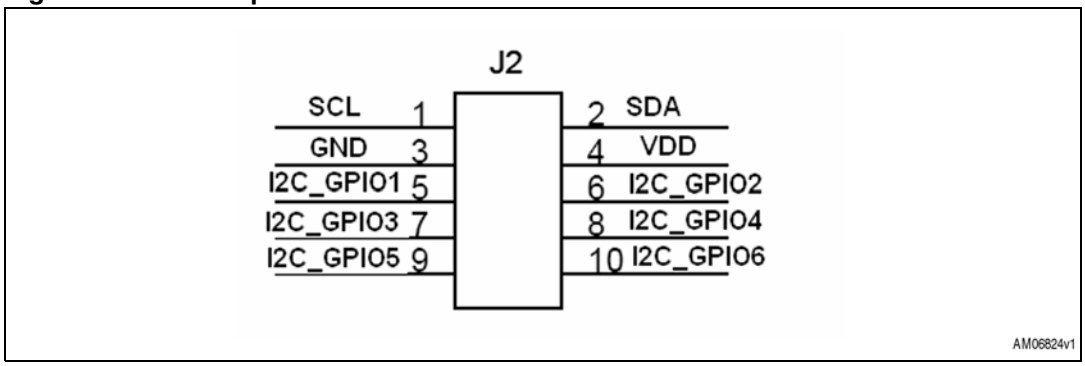
Once this is done, the user can use their own customized GUI to connect to the board.

The user can create their own GUI using the DLLs provided in the package along with the board. The DLL help file is also available along with the package.

3.1 Using the I²C interface of the 10-pin header

Select the I²C interface by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in I²C mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done. *Figure 10* shows the interpretation of the 10-pin header when it is configured in I²C mode.

Figure 10. J1 Interpretation for I²C interface

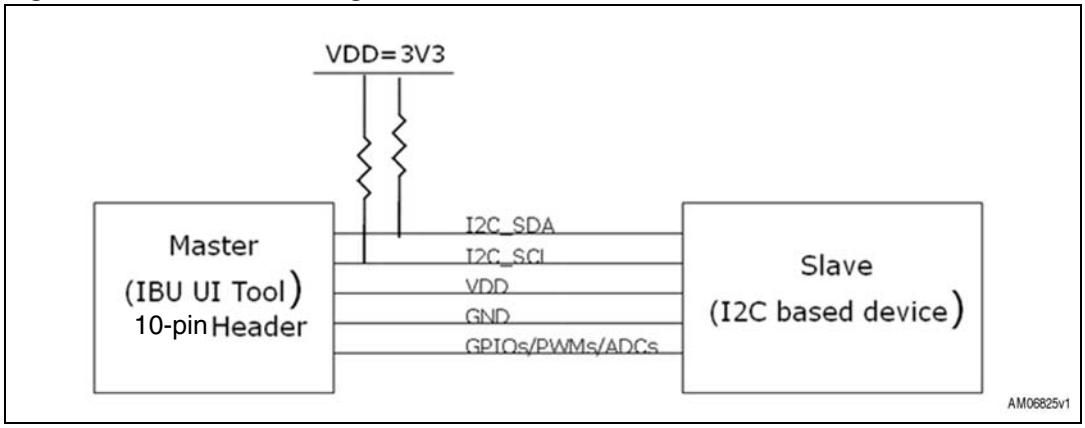


3.1.1 Steps for making the hardware connection

To use any I²C based slave with the IBU UI tool, you need to make the connection for jumper J1, as shown in *Figure 11*.

- The SCL (synchronous clock line), SDA (serial data), and GND (ground line) should be connected to the corresponding lines of the daughter board for I²C communication
- VDD (power supply line) of the two boards should be connected if the daughter board is to be powered using the IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements
- As shown in *Figure 11* below, the SDA and SCL line of the interface is already pulled up to 3.3 V through a resistive pull-up of value 4.7 kΩ.

Figure 11. Connection diagram for I²C interface/GPIOs



3.1.2 GPIO settings

For the GPIO which is to be used along with the I^2C interface, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on one pin. Therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 6*.

By default, I2C_GPIO1 to I2C_GPIO6 are in input pull-up mode.

Here you can set only the GPIOs mentioned. I^2C lines (SDA and SCL) and power lines are fixed. To perform the settings of a GPIO, use the I^2C DLL referring to the DLL help file available.

Through selection, the GPIO can be set in different modes, as shown in *Table 7*, such as simple input mode, input with interrupt, and push-pull output mode. Also in the I^2C interface, there is an option in GPIO5 and GPIO6 to use this GPIO as the PWM clock signal and there is an option in GPIO6 to use it as an ADC channel. Please refer to *Table 7* in *Appendix D*.

3.1.3 Using GPIOs in PWM mode settings

As mentioned above, GPIO 5 and GPIO 6 can also additionally be set in PWM mode. To do this, set GPIO5 or GPIO6 in PWM mode and provide the PWM frequency (maximum value tested is around 10 MHz), and also the duty cycle to generate different kinds of clocks.

The frequency of the PWM clock generated can vary from 10 kHz to 10 MHz. The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with the duty cycle at 0 % and logic '1' is obtained with the duty cycle at 100 %. Please note that the PWM generated on GPIO5 and GPIO6 shares the same frequency but can have different duty cycles. Please refer to *Figure 34* in *Appendix D*.

3.1.4 Using GPIOs as ADC mode settings

As mentioned above, GPIO 6 can also additionally be set as analog channel input. For that set the GPIO 6 in ADC mode and do the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Then Analog input can be provided on this pin and the set of the digital value can be obtained. If the resolution set is 8 bit, one byte is obtained for every sample of the ADC conversion. If the resolution set is 12 bit, two bytes are obtained for every sample of the ADC conversion. ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

3.1.5 I^2C read and write operation

Once the I^2C settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

To read/write in the register, select the register address length depending on the slave device.

The I^2C register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123.

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the I^2C communication taking place between the IBU UI board and the I^2C slave daughter board can be checked. Please note that the number of bytes to be written should be non-zero and in decimal format.

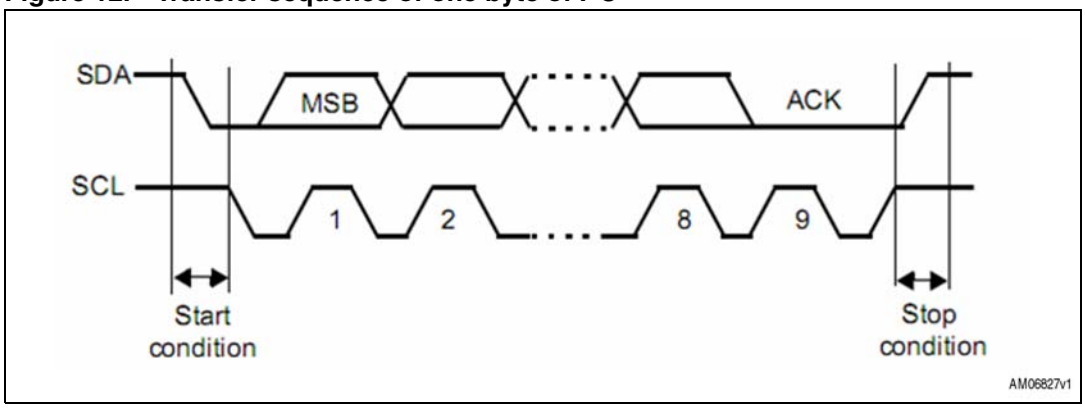
The status messages are of the following types depending on the communication that has taken place:

- Communication complete/bus free
- Wrong acknowledge failure/connection errors
- I^2C timeout that occurs when the slave device does not respond for a predefined interval of time
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED D1 available on the board. It lights up whenever there is any type of communication error of type 2), 3), or 4) above. The LED status is updated after each read or write operation.

The transfer sequence for one byte of I^2C is shown in *Figure 12*.

Figure 12. Transfer sequence of one byte of I^2C

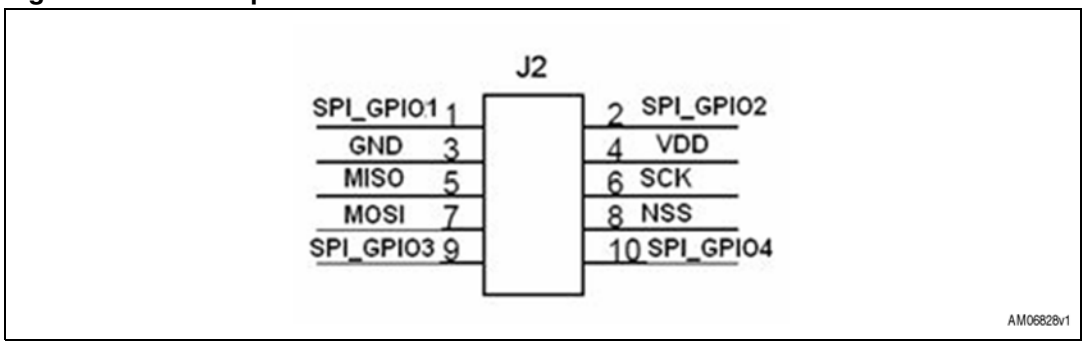


Therefore this interface allows any I^2C interface based slave device to be connected and tested.

3.2 Using the SPI interface of the 10-pin connector

To use the SPI interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in SPI mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done. *Figure 13* below shows the interpretation of the 10-pin header when it is configured in SPI mode.

Figure 13. J2 interpretation for SPI interface

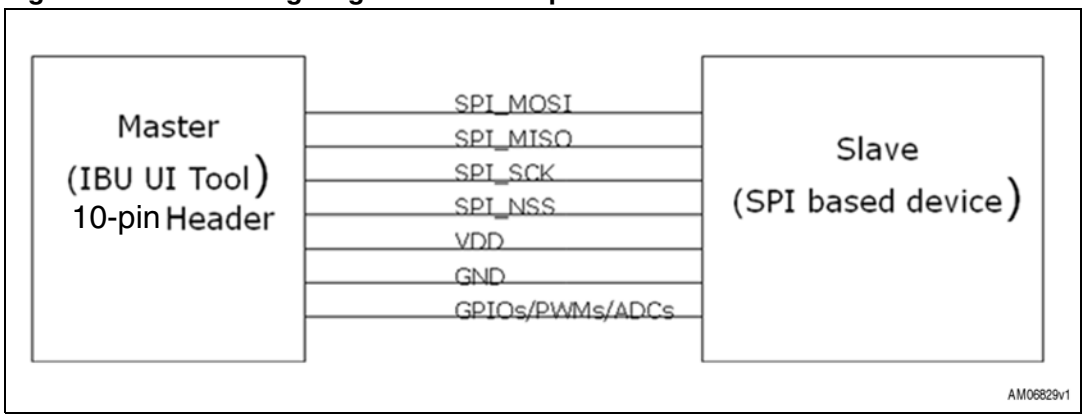


3.2.1 Steps for making hardware connection

To use any SPI based slave with the IBU UI tool, you need to make the connection for jumper J1, as shown in *Figure 14*.

- The SCK (synchronous clock line), MISO (master in slave out), MISO (master out slave in), NSS (slave select) and GND (ground line) should be connected to the corresponding lines of the daughter board for SPI communication
- VDD (power supply line) of the two boards should be connected if the daughter board is to be powered using the IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements

Figure 14. Connecting diagram for the 10-pin SPI interface/GPIOs



3.2.2 GPIO settings

For the GPIO which is to be used along with the SPI interface, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on one pin. Therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 5*.

By default SPI_GPIO1 to SPI_GPIO4 are in input pull-up mode.

Here you can set only the GPIOs mentioned. SPI lines (MISO, MOSI, NSS and SCK) and power lines are fixed. To make the settings of a GPIO, use the SPI DLL referring to the DLL help file available.

Through selection, the GPIO can be set in different modes (as shown in *Table 7*), such as simple input mode, input with interrupt, and push-pull output mode. Also in the SPI interface, there is an option in GPIO3 and GPIO4 to use this GPIO as the PWM clock signal. And there is an option in GPIO4 to use it as the ADC channel.

3.2.3 Using GPIOs as PWM settings

As mentioned above, GPIO 3 and GPIO 4 can also additionally be set in PWM mode. To do this, set the GPIO3 or GPIO4 in PWM mode and provide the PWM frequency (maximum value tested is around 10 MHz) and also the duty cycle to generate different kinds of clocks.

The frequency of the PWM clock generated can vary from 10 kHz to 10 MHz.

The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with duty cycle 0 % and logic '1' is obtained with duty cycle 100 %. Please, refer to *Figure 34*.

3.2.4 Using GPIOs as ADC settings

As mentioned above, GPIO 4 can also additionally be set as an analog channel input. To do this, set the GPIO 4 in ADC mode and perform the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Analog input can then be provided on this pin and the set of the digital value can be obtained.

If the resolution set is 8-bit, one byte is obtained for every sample of the ADC conversion.

If the resolution set is 12-bit, two bytes are obtained for every sample of the ADC conversion. ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

3.2.5 SPI header settings

Once the GPIO settings are done, the daughter board can be connected to the IBU UI board. Before using the SPI communication, some parameters must first be defined.

These parameters include the selection of CPOL, CPHA and baud rate pre-scalar (by default, the most significant bit is put first).

As in SPI standard protocol, CPHA and CPOL values can be 0 or 1. SPI baud rate should be set with values equal to 2, 4, 8, 16, 32, 64, 128, and 256. The SPI base frequency is 36 kHz. Therefore, if the baud rate pre-scalar is set as 4, the SPI runs at a frequency equal to 9 kHz.

Once the selection is made, it sets the SPI interface and now the system is ready to read or write the data from the SPI slave device connected to the IBU UI board.

3.2.6 SPI read and write operation

Once the SPI settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

To read/write in the register, select the register address length depending on the slave device.

The SPI register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123.

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the SPI communication taking place between the IBU UI board and the SPI slave daughter board can be checked. Please note that the number of bytes to be written should be non-zero and in decimal format. The status messages are of the following types depending on the communication that has taken place:

- Communication complete/bus free
- SPI timeout
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED available on the board. It lights up whenever there is any type of communication error of type 2) or 3) above. The LED status is updated after each read or write operation.

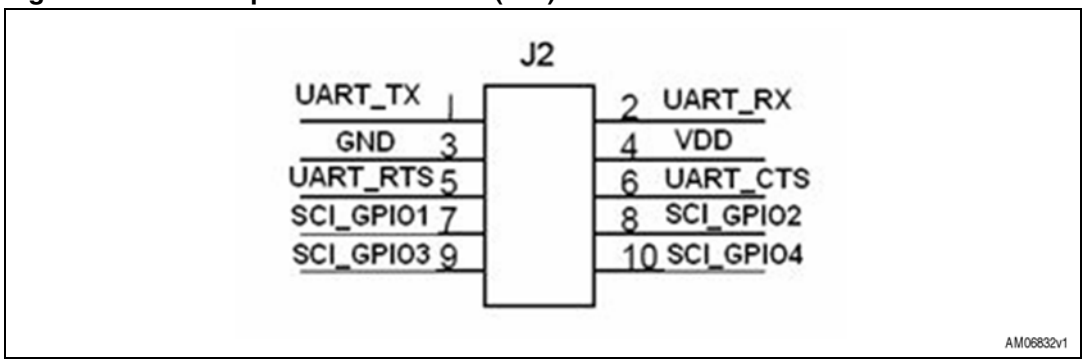
Depending on the CPHA and CPOL values, the transfer sequence for one byte of SPI is shown in *Figure 17*.

Therefore this interface allows any SPI interface based slave device to be connected and tested.

3.3 Using the UART(SCI) interface of the 10-pin header

To use the UART1 interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in UART1 mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done. *Figure 15* below shows the interpretation of the 10-pin headers when it is configured in UART mode.

Figure 15. J1 interpretation for UART (SCI) interface

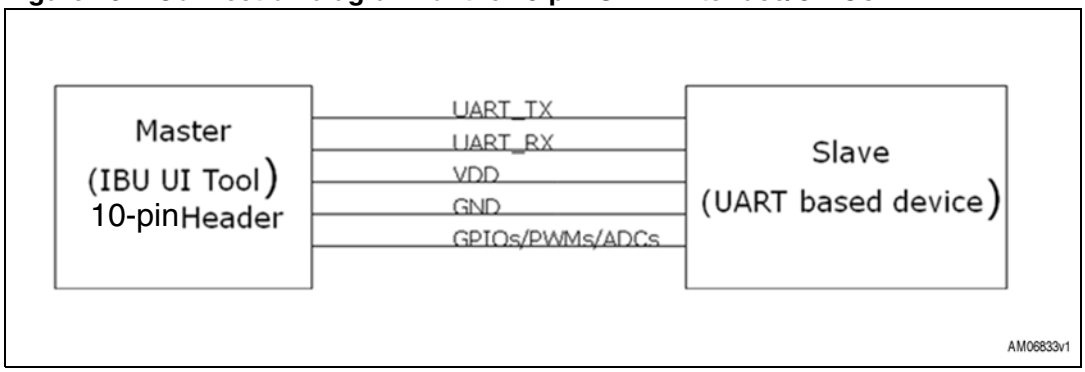


3.3.1 Steps for making hardware connection

To use any UART (SCI) based slave with the IBU UI tool, you need to make the connection for jumper J1, as shown in *Figure 16*.

- The TX (transmitter), RX (receiver), and GND (ground line) should be connected to the corresponding lines of the daughter board for UART (SCI) communication
- VDD equal to 3.3 V (power supply line) of the two boards should be connected if the daughter board is to be powered using the IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements.

Figure 16. Connection diagram for the 10-pin UART interface/GPIOs



3.3.2 GPIO settings

For the GPIO which is to be used along with the UART (SCI) interface, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on two pins, therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 5*.

By default UART_GPIO1 to UART_GPIO4 are in input pull-up mode.

Here you can set only the GPIOs mentioned. UART (TX, RX, CTS, and RTS) lines and power lines are fixed. To perform the settings of a GPIO, use the UART (SCI) DLL referring to the DLL help file available.

Through selection, the GPIO can be set in different modes, such as simple input mode, input with interrupt, and push pull output mode. Also in the UART (SCI) interface, there is an option in GPIO3 and GPIO4 to use this GPIO as the PWM clock signal and there is an option in GPIO4 to use it as an ADC channel.

3.3.3 Using GPIOs in PWM settings

As mentioned above, GPIO 3 and GPIO 4 can also additionally be set in PWM mode. To do this, set the GPIO3 or GPIO4 in PWM mode and provide the PWM frequency (maximum value tested is around 10 MHz) and also the duty cycle to generate different kinds of clocks.

The frequency of the PWM clock generated can vary from 10 kHz to 10MHz. The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with the duty cycle at 0 % and logic '1' is obtained with the duty cycle at 100 %. Please refer to *Figure 34*.

3.3.4 Using GPIO in ADC settings

As mentioned above, GPIO 4 can also additionally be set as analog channel input. To do this, set the GPIO 4 in ADC mode and perform the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Analog input can then be provided on this pin and the set of the digital value can be obtained.

If the resolution set is 8-bit, one byte is obtained for every sample of the ADC conversion.

If the resolution set is 12-bit, two bytes are obtained for every sample of the ADC conversion.

ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

3.3.5 UART1 (SCI1) header settings

Once the GPIO settings are completed, the daughter board can be connected to the IBU UI board. Before using the UART1 (SCI1) communication, some parameters must first be defined.

These parameters include the selection of parameters such as:

- UART (SCI) bits per second values can be 110, 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, and 460800
- UART (SCI) data per bits can be 8-bit or 9-bit
- UART (SCI) parity bits can be even, odd, or none
- UART (SCI) stop bits can be 1 or 2
- UART (SCI) flow control can be hardware, hardware CTS, hardware RTS, or none

Once the selection is made, it sets the UART1 (SCI1) interface and now the system is ready to read or write the data from the UART1 (SCI1) slave device connected to the IBU UI board.

3.3.6 UART1 (SCI1) read and write operation

Once the UART1 (SCI1) settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

To read/write in the register, select the register address length depending on the slave device.

The UART1 (SCI1) register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123.

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the UART1 (SCI1) communication taking place between the IBU UI board and the UART1 (SCI1) slave daughter board can be checked. Please note that the number of bytes to be written should

be non-zero and in decimal format. The status messages are of the following types depending on the communication that has taken place.

- Communication complete/bus free
- Error conditions
- UART(SCI) timeout
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED available on the board. It lights up whenever there is any type of communication error of type 2), 3), or 4) above. The LED status is updated after each read or write operation. Depending on the stop bits and data per bits configured, the transfer sequence for one byte of UART (SCI) is shown in *Figure 25*.

Therefore this interface allows any UART (SCI) interface based slave device to be connected and tested.

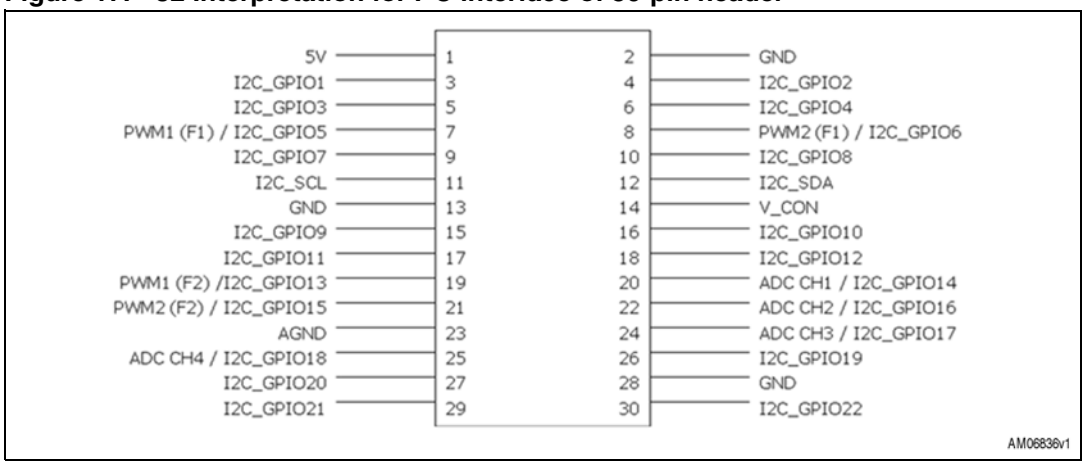
3.4 Using the I²C interface of the 30-pin header

To use the I²C interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in I²C mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done.

Figure 17 below shows the interpretation of the 30-pin header when it is configured in UART mode.

Along with the I²C communication interface, the 30-pin header in I²C interface mode also consists of 22 configurable GPIOs in various modes. Of these 22 GPIOs, 4 can additionally be configured as analog channels and another 4 GPIOs can be configured as PWM channels.

Figure 17. J2 Interpretation for I²C interface of 30-pin header

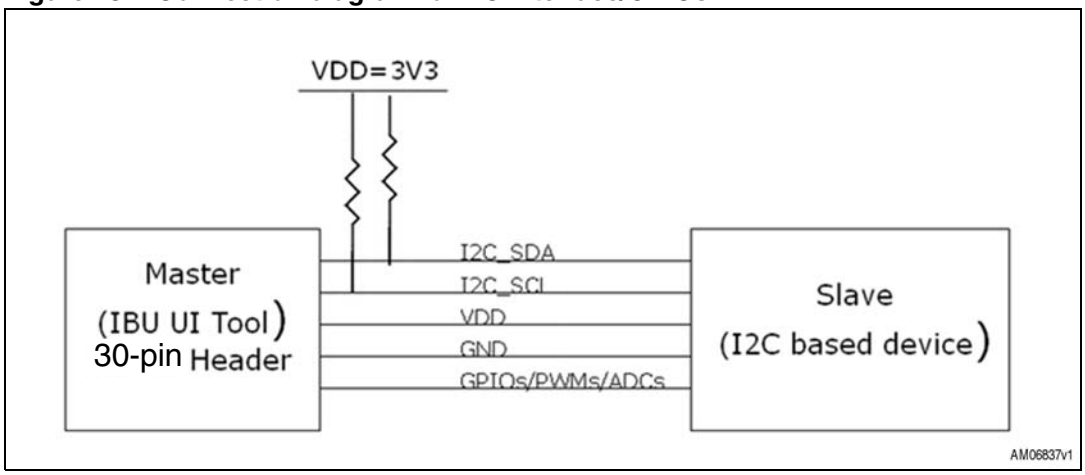


3.4.1 Steps for making hardware connection

To use any I²C based slave with the IBU UI tool, you need to make the connection for jumper J2, as shown in *Figure 18*.

- The SCL (synchronous clock line), SDA (serial data), and GND (ground line) should be connected to the corresponding lines of the daughter board for I²C communication
- VDD (power supply line) of the two boards should be connected if the daughter board is to be powered using the IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements
- As shown in *Figure 18*, the SDA and SCL line of the interface is already pulled up to 3.3 V through a resistive pull up value of 4.7 k.

Figure 18. Connection diagram for I²C interface/GPIOs



3.4.2 GPIO settings

For the GPIO which is to be used along with the I²C interface of the 30-pin header, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on 4 pins, therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 6*.

Through selection, the GPIO can be set in different modes, as shown in *Table 8*.

By default I2C_GPIO1 to I2C_GPIO22 are in input pull-up mode.

3.4.3 Using GPIOs in PWM mode settings

Also in 30 pin interface in I²C mode, there is provision to use pin# 7,8,19,21 to use these GPIOs as PWM clock signal. As shown in the table below, PWM channel 1 is available on pin 7 of the 30-pin interface. Please refer to *Table 9*.

Note: PWM channel 1 and 2 can have different duty cycles but they share the same frequency

To generate different kinds of clocks, configure the PWM channel (1, 2, 3, or 4) by providing the PWM frequency (maximum value tested is around 10 MHz) and the duty cycle. Please refer to *Figure 34* for the PWM signal.

The frequency of the PWM clock generated can vary from 10 kHz to 10 MHz.

The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with the duty cycle at 0 % and logic '1' is obtained with the duty cycle at 100 %.

3.4.4 Using GPIOs in ADC mode settings

There are 4 ADC channels available on pin# 20, 22, 24, and 25 of the 30-pin interface. So these pins can also be additionally set as analog channel input. To do this, set the particular channel in ADC mode and perform the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Analog input can then be provided on this pin and the set of the digital value can be obtained.

If the resolution set is 8-bit, one byte is obtained for every sample of the ADC conversion.

If the resolution set is 12-bit, two bytes are obtained for every sample of the ADC conversion.

ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

ADC resolution values can be 8-bit or 12-bit.

3.4.5 I^2C header settings

Once the GPIO settings have been made, the daughter board can be connected to the IBU UI board. Before using the I^2C communication, some parameters must first be defined.

These parameters include the selection of I^2C address types (7-bit or 10-bit), I^2C slave device address, and I^2C speed.

As in I^2C standard protocol, the I^2C address type can be 7-bit or 10-bit addressing. The I^2C address is a one byte address in the case of 7-bit addressing and a 2 byte address in the case of 10-bit addressing. The I^2C speed should be set between 10 kHz to 400 kHz.

Once the selection is made, it sets the I^2C interface and now the system is ready to read or write the data from the I^2C slave device connected to the IBU UI board.

3.4.6 I^2C read and write operation

Once the I^2C settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

To read/write in the register, select the register address length depending on the slave device.

The I^2C register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123.

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the I^2C communication taking place between IBU UI board and the I^2C slave daughter board can be

checked. Please note that the number of bytes to be written should be non-zero and in decimal format.

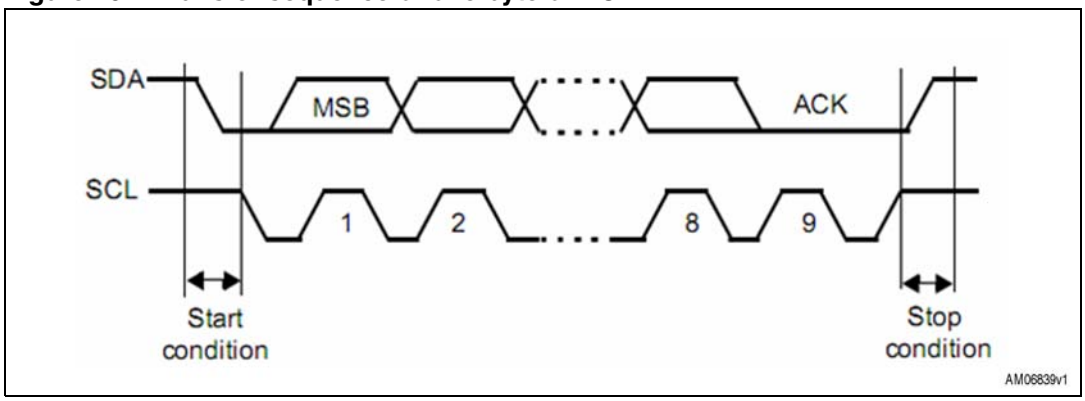
The status messages are of the following types depending on the communication that has taken place.

- Communication complete/bus free
- Wrong acknowledge failure/connection errors
- I^2C timeout
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED available on the board. It lights up whenever there is any type of communication error of type 2), 3), or 4) above. The LED status is updated after each read or write operation.

The transfer sequence for one byte of I^2C is shown in *Figure 19*.

Figure 19. Transfer sequence of one byte of I^2C



Therefore, this interface allows any I^2C interface based slave device to be connected and tested.

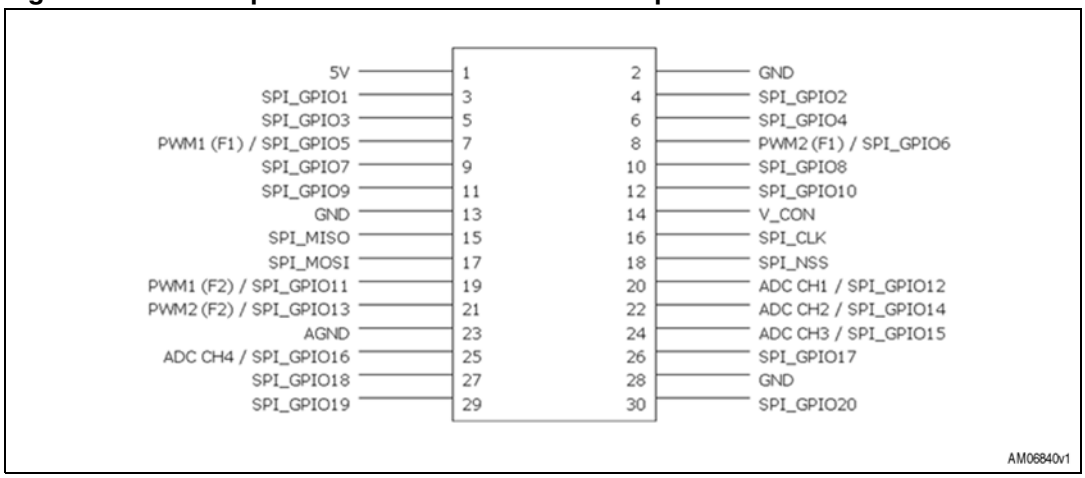
3.5 Using the SPI interface of the 30-pin header

To use the SPI interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in SPI mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done.

Along with the SPI communication interface, the 30-pin header in SPI interface mode also consists of 20 configurable GPIOs in various modes. Of these 20 GPIOs, 4 can additionally be configured as analog channels and another 4 GPIOs can be configured as PWM channels.

Figure 20 below shows the interpretation of the 30-pin header when it is configured in SPI mode.

Figure 20. J2 Interpretation for SPI interface of 30-pin header

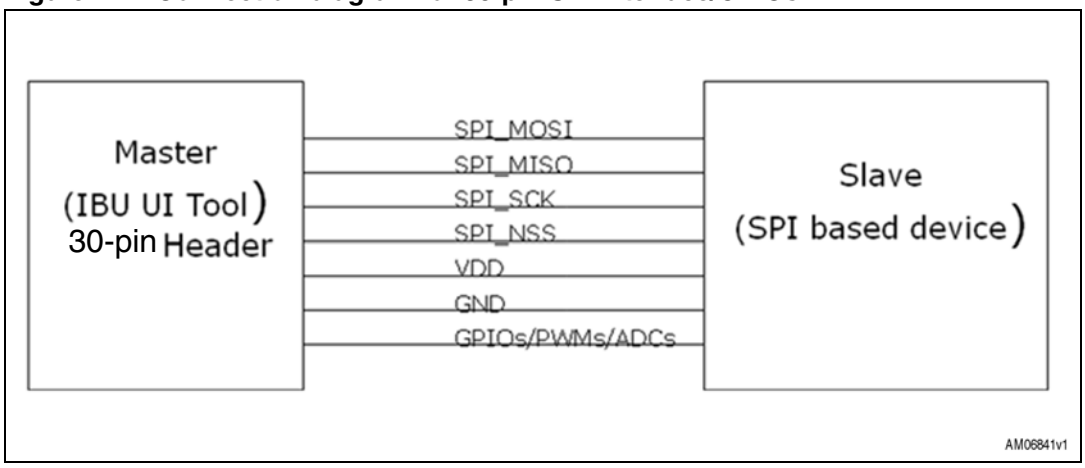


3.5.1 Steps for making hardware connection

To use any SPI based slave with the IBU UI tool, you need to make the connection for jumper J2, as shown in *Figure 21*.

- The SCK (synchronous clock line), MISO (master in slave out), MISO (master out slave in), NSS (slave select), and GND (ground line) should be connected to the corresponding lines of the daughter board for SPI communication
- VDD (power supply line) of the two boards should be connected if the daughter board is to be powered using IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements.

Figure 21. Connection diagram for 30-pin SPI interface/GPIOs



3.5.2 GPIO settings

For the GPIO which is to be used along with the SPI interface of the 30-pin header, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on 4 pins, therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 6*. Through selection, the GPIO can be set in different modes, as shown in *Table 8*.

By default SPI_GPIO1 to SPI_GPIO20 are in input pull-up mode. Please refer to *Table 8*.

3.5.3 Using GPIOs in PWM mode settings

Also in the 30-pin interface in SPI mode, there is a provision to use pin# 7, 8, 19, and 21 to use these GPIOs as the PWM clock signal. PWM channel 1 is available on pin 7 of the 30-pin interface. Please refer to *Table 9*.

Note: PWM channel 1 and 2 can have different duty cycles but they share the same frequency.

To generate different kinds of clocks, configure the PWM channel (1, 2, 3, or 4) by providing the PWM frequency (maximum value tested is around 10 MHz) and the duty cycle. Please, refer to*Figure 34*.

The frequency of the PWM clock generated can vary from 10 kHz to 10 MHz.

The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with the duty cycle at 0 % and logic '1' is obtained with the duty cycle at 100 %.

3.5.4 Using GPIOs in ADC mode settings

There are 4 ADC channels available on pin# 20, 22, 24, and 25 of the 30-pin interface. So these pins can also be additionally set as analog channel input. To do this, set the particular channel in ADC mode and perform the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Analog input can then be provided on this pin and the set of the digital value can be obtained.

If the resolution set is 8-bit, one byte is obtained for every sample of the ADC conversion. If the resolution set is 12-bit, two bytes are obtained for every sample of the ADC conversion.

ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

ADC resolution values can be 8-bit or 12-bit.

3.5.5 SPI header settings

Once the GPIO settings have been made, the daughter board can be connected to the IBU UI board. Before using the SPI communication, some parameters must first be defined.

These parameters include the selection of CPOL, CPHA and baud rate pre-scalar.

By default, the most significant bit is put first.

As in SPI standard protocol, CPHA and CPOL values can be 0 or 1. SPI baud rate should be set with values equal to 2, 4, 8, 16, 32, 64, 128, or 256. The SPI base frequency is 36 kHz. So, if the baud rate pre-scalar is set as 4, the SPI runs at a frequency equal to 9 kHz.

Once the selection is made, it sets the SPI interface and now the system is ready to read or write the data from the SPI slave device connected to the IBU UI board.

3.5.6 SPI read and write operation

Once the SPI settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

To read/write in the register, select the register address length depending on the slave device.

The SPI register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123.

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the SPI communication taking place between the IBU UI board and the SPI slave daughter board can be checked. Please note that the number of bytes to be written should be non-zero and in decimal format.

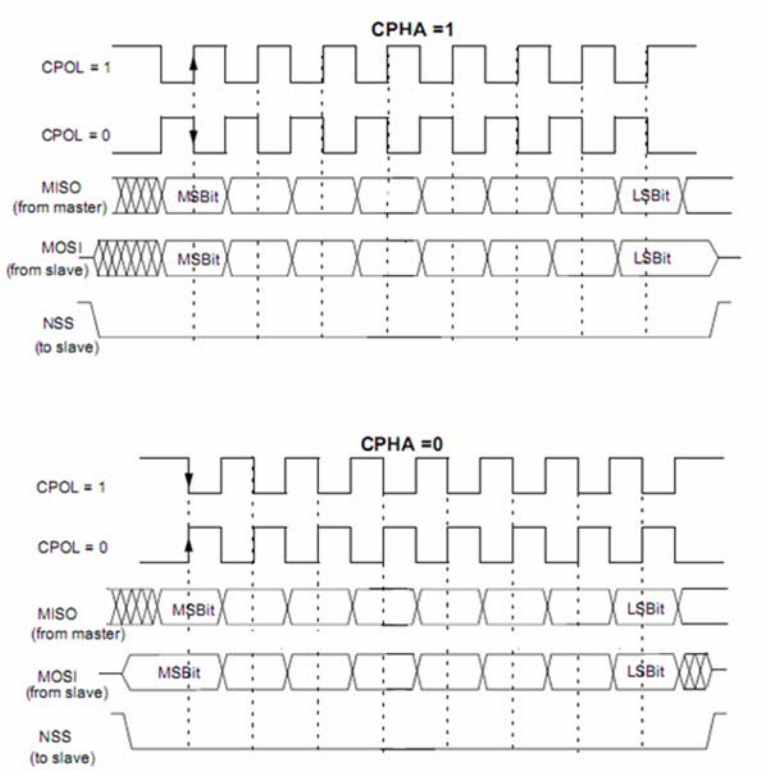
The status messages are of the following types depending on the communication that has taken place:

- Communication complete/bus free
- SPI timeout
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED available on the board. It lights up whenever there is any type of communication error of type 1), 2), or 3) above. The LED status is updated after each read or write operation.

The transfer sequence for one byte of SPI is shown in *Figure 22*.

Figure 22. Transfer sequence of one byte of SPI



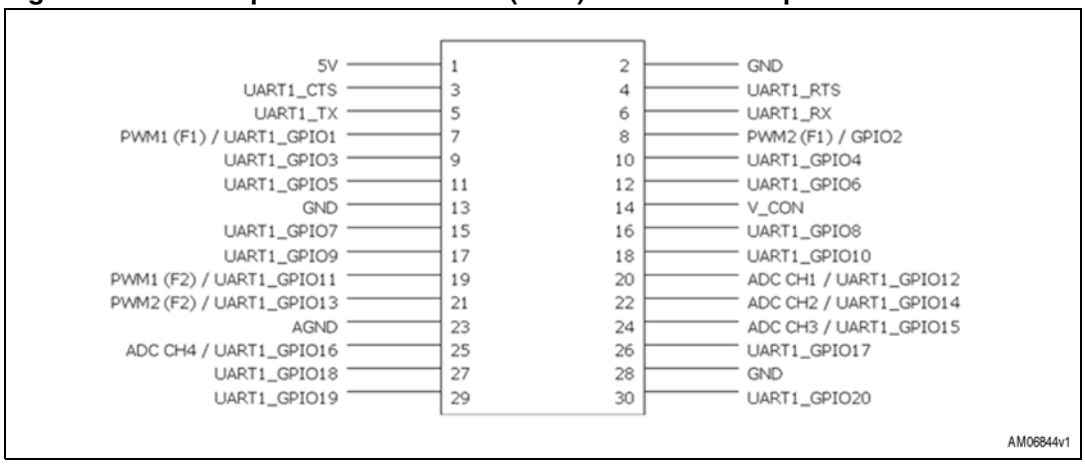
Therefore, this interface allows any SPI interface based slave device to be connected and tested.

3.6 Using the UART1(SCI1) interface of the 30-pin header

To use the UART1 interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in UART1 mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done. *Figure 23* below shows the interpretation of the 30-pin header when it is configured in UART1 mode.

Along with the UART1 (SCI1) communication interface, the 30-pin header in UART1 (SCI1) interface mode also consists of 20 configurable GPIOs in various modes. Of these 20 GPIOs, 4 can additionally be configured as analog channels and another 4 GPIOs can be configured as PWM channels.

Figure 23. J2 interpretation for UART1 (SCI1) interface of 30-pin header

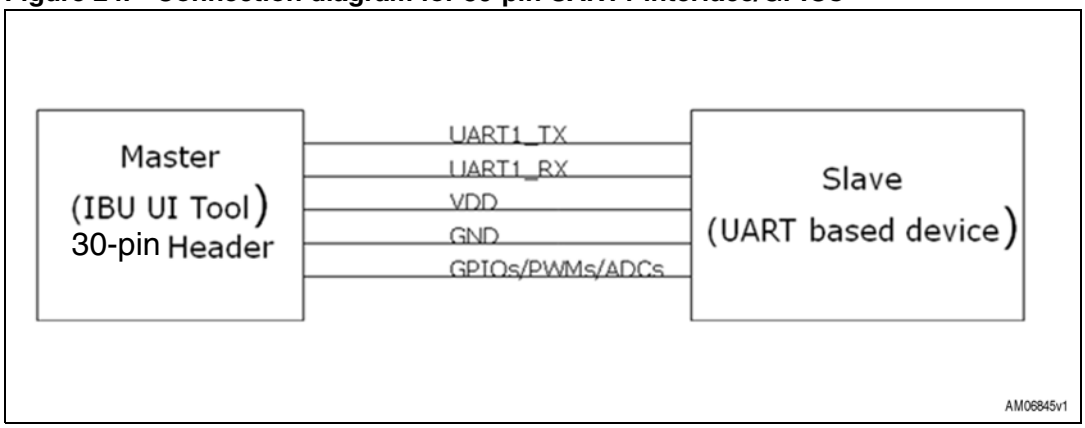


3.6.1 Steps for making hardware connection

To use any UART1 (SCI1) based slave with the IBU UI tool, you need to make the connection for jumper J2, as shown in *Figure 24*.

- The SCL (synchronous clock line), SDA (serial data), and GND (ground line) should be connected to the corresponding lines of the daughter board for UART1 (SCI1) communication
- VDD (power supply line) of the two boards should be connected if the daughter board is to be powered using the IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements.

Figure 24. Connection diagram for 30-pin UART1 interface/GPIOs



3.6.2 GPIO settings

For the GPIO which is to be used along with the UART1 (SCI1) interface of the 30-pin header, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on 4 pins, therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 6*.

Through selection, the GPIO can be set in different modes. By default, UART1 (SCI1)_GPIO1 to UART1 (SCI1)_GPIO20 are in input pull-up mode.

3.6.3 Using GPIOs in PWM mode settings

Also in the 30-pin interface in UART1 (SCI1) mode, there is a provision to use pin# 7,8, 19, and 21 to use these GPIOs as PWM clock signal. The PWM channel 1 is available on pin 7 of the 30-pin interface. Please refer to *Table 9*.

Note: PWM channel 1 and 2 can have different duty cycles but they share the same frequency

To generate different kinds of clocks, configure the PWM channel (1, 2, 3, or 4) by providing the PWM frequency (maximum value tested is around 10 MHz) and the duty cycle. Please refer to *Figure 34*.

The frequency of the PWM clock generated can vary from 10 kHz to 10 MHz.

The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with the duty cycle at 0 % and logic '1' is obtained with the duty cycle at 100 %.

3.6.4 Using GPIOs in ADC mode settings

There are 4 ADC channels available on pin# 20, 22, 24, and 25 of the 30-pin interface. So these pins can also be additionally set as analog channel input. To do this, set the particular channel in ADC mode and perform the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Analog input can be provided on this pin and the set of the digital value can be obtained.

If the resolution set is 8-bit, one byte is obtained for every sample of the ADC conversion.

If the resolution set is 12-bit, two bytes are obtained for every sample of the ADC conversion.

ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

3.6.5 UART1 (SCI1) header settings

Once the GPIO settings are completed, the daughter board can be connected to the IBU UI board. Before using the UART1 (SCI1) communication, some parameters must first be defined.

These parameters include the selection of parameters such as:

- UART (SCI) bits per second values can be 110, 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, and 460800
- UART (SCI) data per bits can be 8-bit or 9-bit
- UART (SCI) parity bits can be even, odd, or none
- UART (SCI) stop bits can be 1 or 2
- UART (SCI) flow control can be hardware, hardware CTS, hardware RTS, or none

Once the selection is made, it sets the UART1 (SCI1) interface and now the system is ready to read or write the data from the UART1 (SCI1) slave device connected to the IBU UI board.

3.6.6 UART1 (SCI1) read and write operation

Once the UART1 (SCI1) settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

To read/write in the register, select the register address length depending on the slave device.

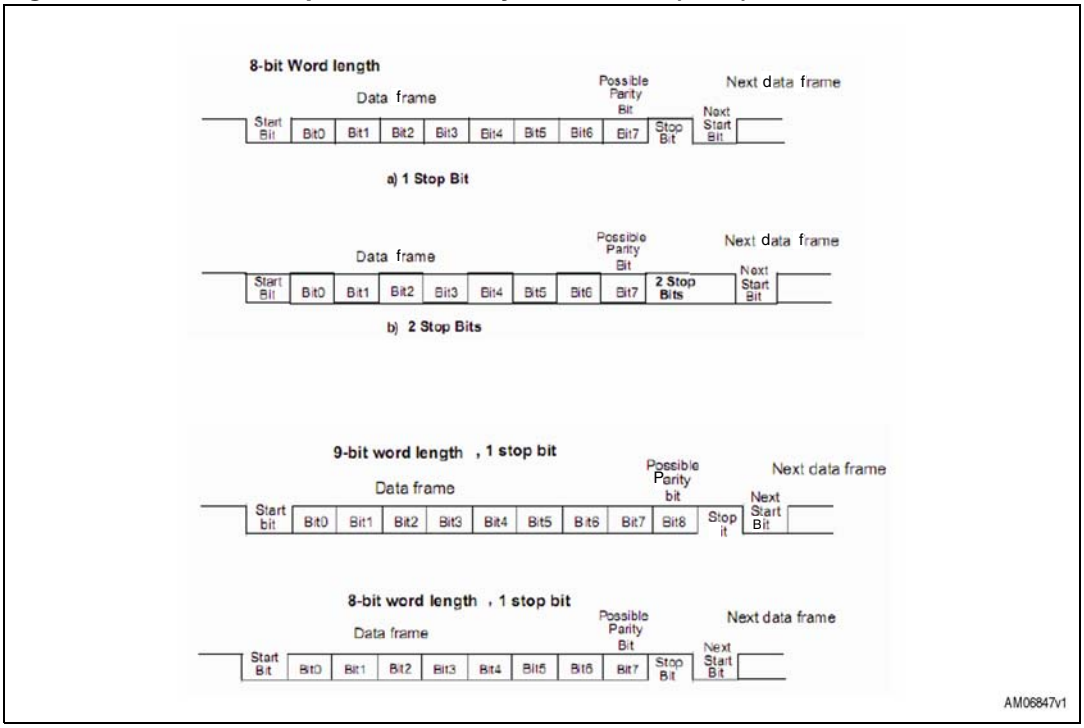
UART1 (SCI1) register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the UART1 (SCI1) communication taking place between IBU UI board and the UART1 (SCI1) slave daughter board can be checked. Please note that the number of bytes to be written should be non-zero and in decimal format. The status messages are of the following types depending on the communication that has taken place.

- Communication complete/bus free
- Error conditions
- UART1 (SCI1) timeout
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED available on the board. It lights up whenever there is any type of communication error of type 2), 3), or 4) above. The LED status is updated after each read or write operation. The transfer sequence for one bye of UART1 (SCI1) is shown in *Figure 25*.

Figure 25. Transfer sequence of one byte of UART1 (SCI1)

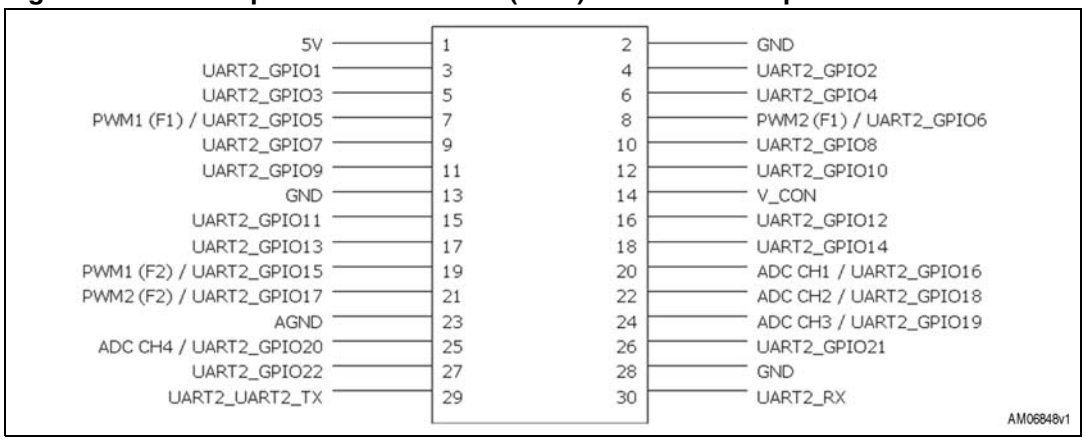


Therefore, this interface allows any UART1 (SCI1) interface based slave device to be connected and tested.

3.7 Using UART2 (SCI2) interface of 30-pin header

To use the UART1 interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in UART1 mode. The section below explains how the tool and its features behave once selection has been made using the DLLs and it also explains how the hardware setup is to be done. *Figure 26* below shows the interpretation of the 30-pin header when it is configured in UART2 mode. Along with the UART2 (SCI2) communication interface, the 30-pin header in UART2 (SCI2) interface mode also consists of 22 configurable GPIOs in various modes. Of these 22 GPIOs, 4 can additionally be configured as analog channels and another 4 GPIOs can be configured as PWM channels.

Figure 26. J2 interpretation for UART2 (SCI2) interface of 30-pin header

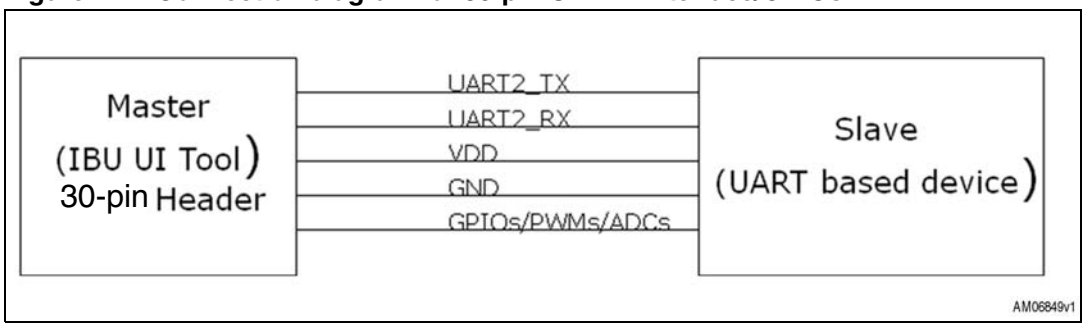


3.7.1 Steps for making hardware connection

To use any UART2 (SCI2) based slave with the IBU UI tool, you need to make the connection for jumper J2, as shown in figure below.

- The UART2_TX and UART2_RX lines and GND (ground line) should be connected to the corresponding lines of the daughter board for UART2 (SCI2) communication
- VDD (power supply line) of the two boards should be connected if the daughter board is to be powered using the IBU UI tool
- The GPIOs of the IBU UI tool and daughter board can be connected or left unconnected as per user requirements.

Figure 27. Connection diagram for 30-pin UART2 interface/GPIOs



3.7.2 Select UART2 (SCI2) interface using DLL software

To use the UART2 (SCI2) interface, it must be selected by sending the command from the DLL, as mentioned in the DLL help file. After this, the board is ready to be used in UART2 (SCI2) mode.

3.7.3 GPIO settings

For the GPIO which is to be used along with the UART2 (SCI2) interface of the 30-pin header, it is necessary to make the proper settings. These GPIOs may be used as control lines, chip select or status line, such as interrupt line, or to generate a clock signal using the PWM feature available on 4 pins, therefore, you need to make the GPIOs settings accordingly.

To understand the modes that are supported by a particular pin, please refer to *Table 6*. Through selection, the GPIO can be set in different modes, as shown in *Table 8*.

By default UART1 (SCI1)_GPIO1 to UART1 (SCI1)_GPIO20 are in Input pull-up mode, please refer to *Table 8*.

3.7.4 Using GPIOs in PWM mode settings

3. Also in the 30-pin interface in UART2 (SCI2) mode, there is a provision to use Pin# 7, 8, 19, and 21, to use these GPIOs as PWM clock signals. PWM channel 1 is available on pin 7 of the 30-pin interface. Please refer to *Table 9*.

Note: PWM channel 1 and 2 can have different duty cycles but they share the same frequency

To generate different kinds of clocks, configure the PWM channel (1, 2, 3, or 4) by providing the PWM frequency (maximum value tested is around 10 MHz) and the duty cycle. Please, refer to *Figure 34*.

The frequency of the PWM clock generated can vary from 10 kHz to 10 MHz.

The duty cycle of the PWM clock can vary from 0 % to 100 %. Logic '0' is obtained with the duty cycle at 0 % and logic '1' is obtained with the duty cycle at 100 %.

3.7.5 Using GPIOs in ADC mode settings

There are 4 ADC channels available on pin# 20, 22, 24, and 25 of the 30-pin interface. So these pins can also be additionally set as analog channel input. To do this, set the particular channel in ADC mode and perform the analog settings. Analog settings include ADC sample time selection and ADC resolution. After that, specify the number of samples that are required. Analog input can then be provided on this pin and the set of the digital value can be obtained.

If the resolution set is 8-bit, one byte is obtained for every sample of the ADC conversion.

If the resolution set is 12-bit, two bytes are obtained for every sample of the ADC conversion.

ADC sample time selection values can be one of the following:

7.5 cycles, 13.5 cycles, 28.5 cycles, 41.5 cycles, 55.5 cycles, 71.5 cycles, or 239.5 cycles.

3.7.6 UART2 (SCI2) header settings

Once the GPIO settings are completed, the daughter board can be connected to the IBU UI board. Before using the UART2 (SCI2) communication, some parameters must first be defined.

These parameters include the selection of parameters such as:

- UART (SCI) bits per second values can be 110, 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, and 460800
- UART (SCI) data per bits can be 8-bit or 9-bit
- UART (SCI) parity bits can be even, odd, or none
- UART (SCI) stop bits can be 1 or 2
- UART (SCI) flow control can be hardware, hardware CTS, hardware RTS, or none

Once the selection is made, it sets the UART2 (SCI2) interface and now the system is ready to read or write the data from the UART2 (SCI2) slave device connected to the IBU UI board.

3.7.7 UART2 (SCI2) read and write operation

Once the UART2 (SCI2) settings have been made, it is possible to read the registers of the slave device and write in the registers of the slave device. After every read and write operation, the user can obtain information about the status of the communication.

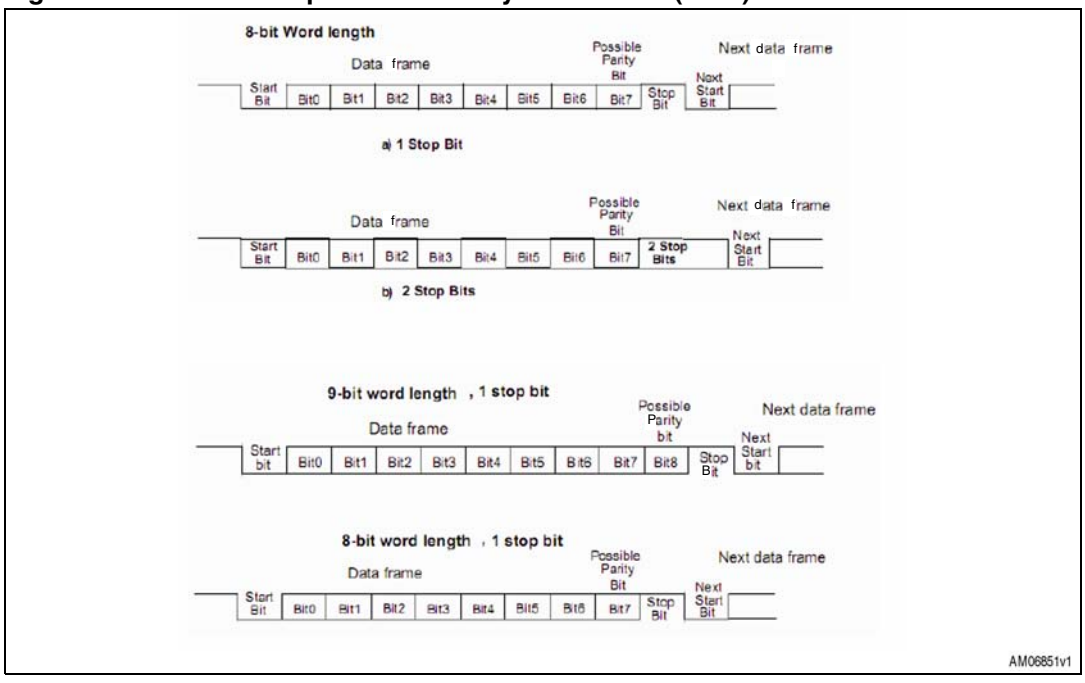
To read/write in the register, select the register address length depending on the slave device. The UART2 (SCI2) register address length can range from 0 to 4 bytes. Depending on the address length given, the register address should be provided in hex format. For instance, if the register address is 3 bytes, its value should be in the form 0x123.

Then, provide values, to read and write from the slave device, to the tool. Data to be written should be provided in the hex format. After every read or write operation, the tool provides the status (e.g. status: communication complete/bus free) so that the status of the UART2 (SCI2) communication taking place between the IBU UI board and the UART2 (SCI2) slave daughter board can be checked. Please note that the number of bytes to be written should be non-zero and in decimal format. The status messages are of the following types depending on the communication that has taken place.

- Communication complete/bus free
- Error conditions
- UART2 (SCI2) timeout
- Other reasons: this occurs when the user tries to perform read/write operations with a data length equal to zero.

Also, there is a communication status LED available on the board. It lights up whenever there is any type of communication error of type 2), 3), or 4) above. The LED status is updated after each read or write operation. The transfer sequence for one bye of UART2 (SCI2) is shown in *Figure 28*.

Figure 28. Transfer sequence of one byte of UART2 (SCI2)



Therefore, this interface allows any UART2 (SCI2) interface based slave device to be connected and tested.

4 Working in DFU mode

To work in DFU mode, please send the appropriate command through the DLL. To do this, please refer to the DLL help file.

The DFU setup is available at www.st.com/mcu/modules.

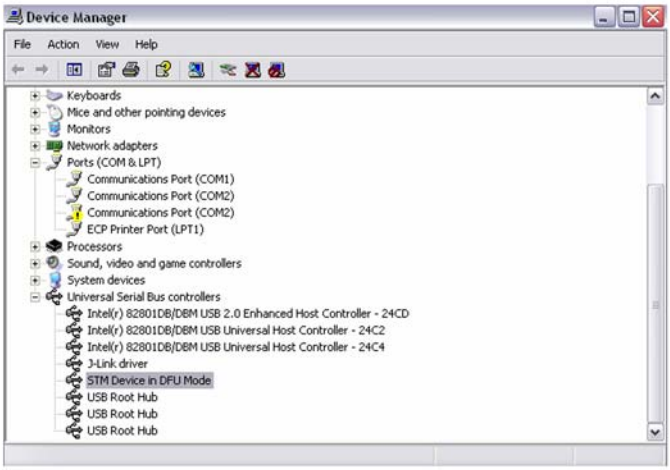
Scroll down to Software-PC\DFUSE on the relevant webpage to download the zip folder.

The folder contains the setup files. After installing the setup, plug in the board. When the PC asks for the driver, browse to the path of the driver. The driver is available at the installed software path at Program Files\STMicroelectronics\DFUSe\Driver.

The user manual for the DFU GUI is also available on the same link.

As a result, you should find the board enumerated as device firmware upgrade and it is shown as “Device Firmware Upgrade”, as seen in *Figure 29*. If this message does not appear, please contact technical support.

Figure 29. Enumeration in DFU mode



Appendix A Schematics and BOM list

Figure 30. Microcontroller section

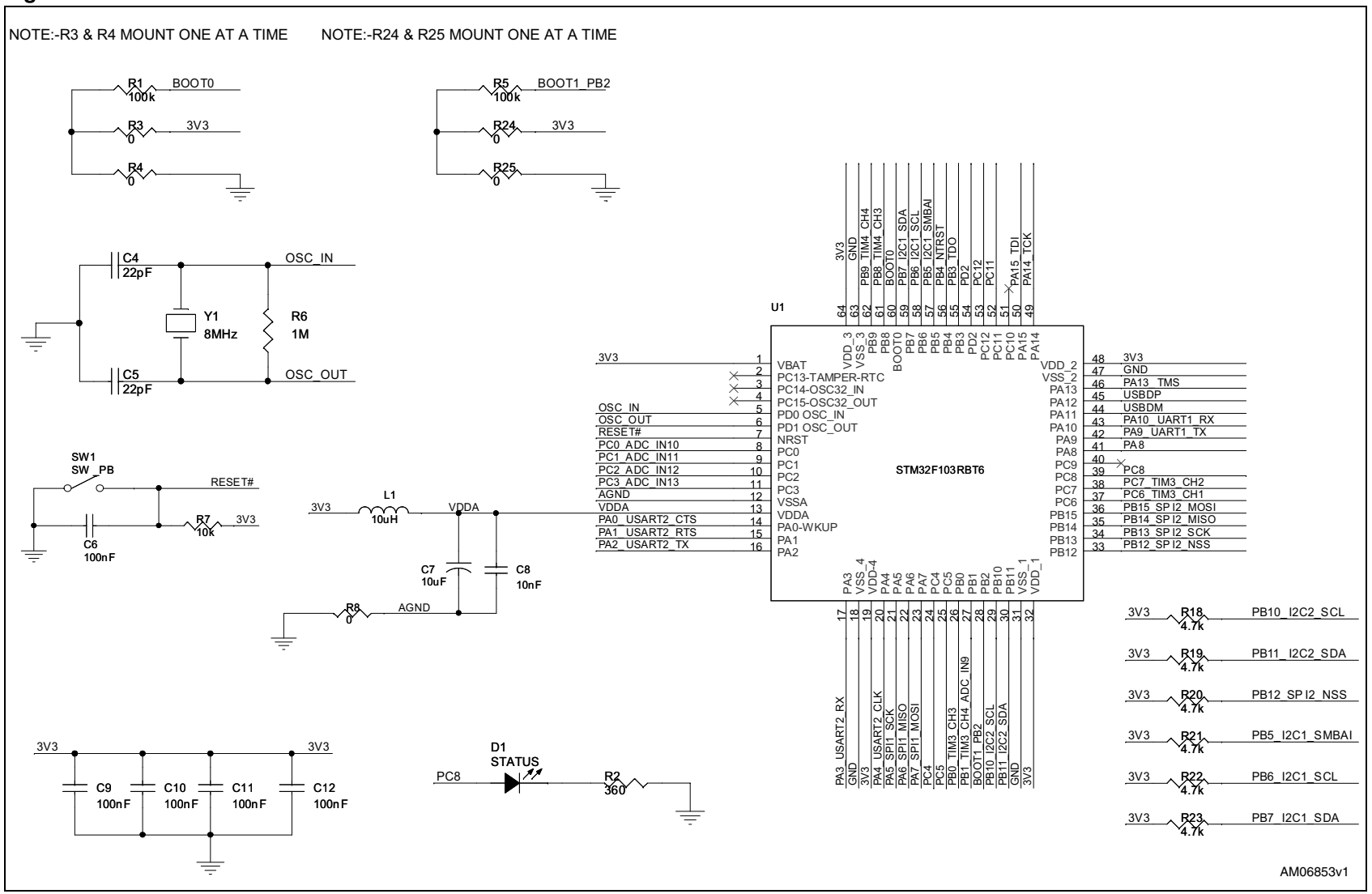


Figure 31. JTAG interface, mode selection switch and power supply section

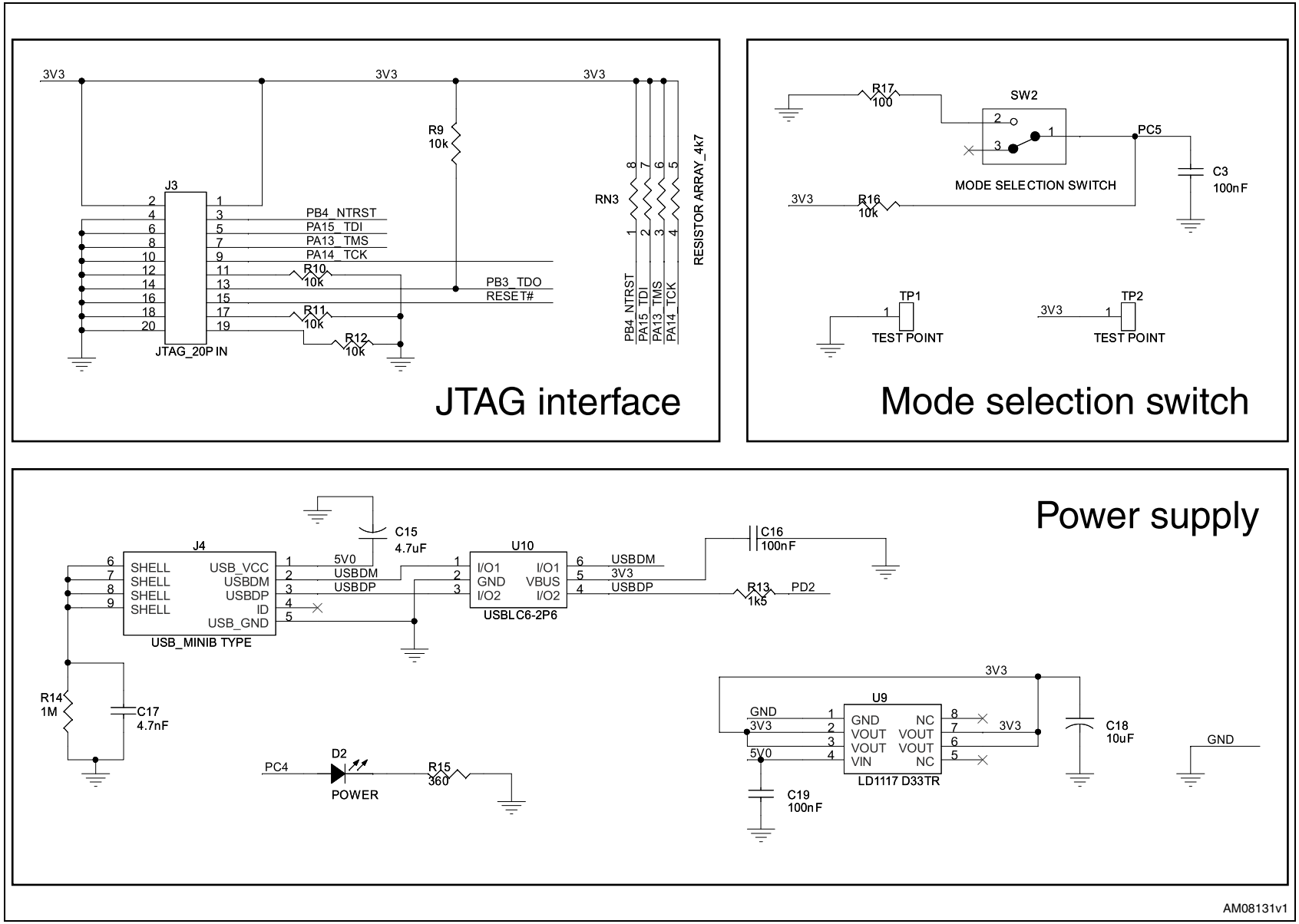


Figure 32. 10-pin com interface

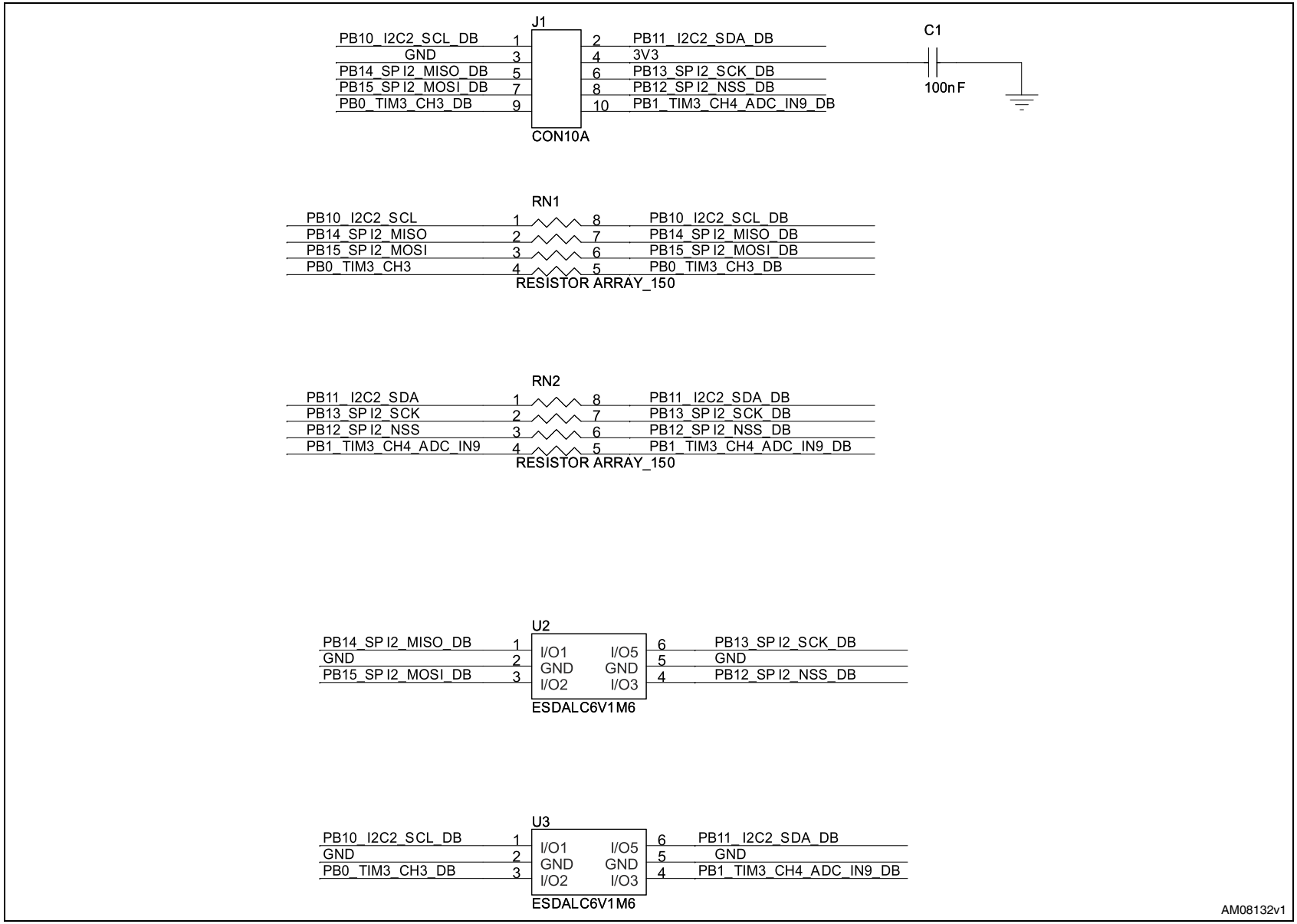


Figure 33. 30-pin com interface

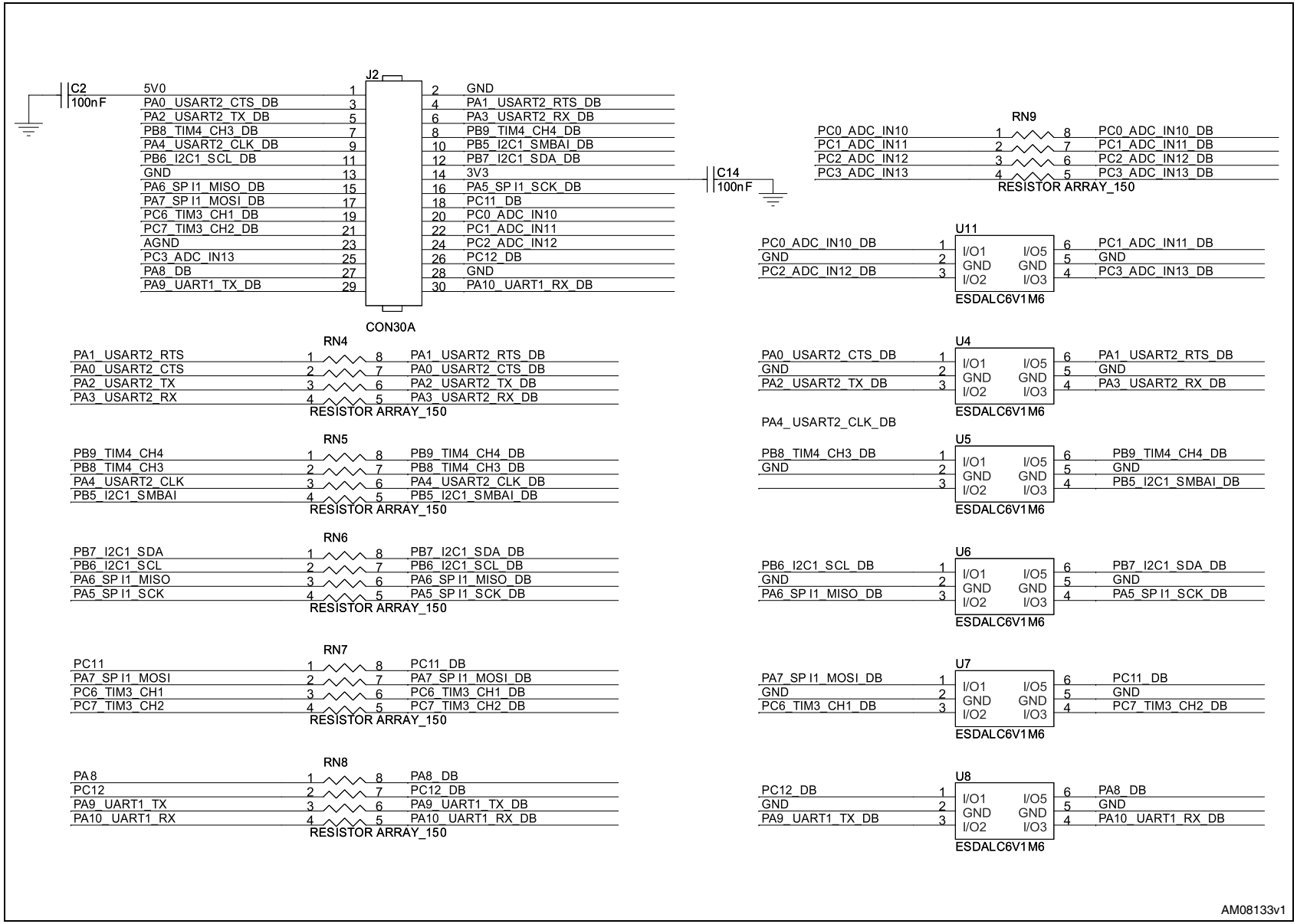


Table 4. BOM

Category	Reference designator	Component Description	Package	Manufacturer	Manufacturer's ordering code / orderable part number	Supplier	Supplier ordering code
ST devices	U9	LD1117D33TR	SO-8	STMicroelectronics	LD1117D33TR	STMicroelectronics	LD1117D33TR
	U1	STM32F103RBT6	LQFP64	STMicroelectronics	STM32F103RBT6	STMicroelectronics	STM32F103RBT6
	U10	USBLC6-2P6	SOT-666	STMicroelectronics	USBLC6-2P6	STMicroelectronics	USBLC6-2P6
	U2,U3,U4,U5, U6,U7,U8, U11	ESDALC6V1M6	uQFN16	STMicroelectronics	ESDALC6V1M6	STMicroelectronics	ESDALC6V1M6
NON ST devices							
Crystal and oscillator	Y1	CRYSTAL 8.00 MHZ 20 pF 49US	11.35 x 4.5 mm crystal			Digi-Key	X1094-ND
Connectors and jumpers	J1	Box header 2.54 mm, double row R/A, 10pin,	Header 2x5 pin, 2.54 mmx2.54 mm Pitch			Protectron	P9604-10-15-1
	J2	Box header 2.54 mm, double row R/A, 30 pin,	Header 2x15 pin, 2.54mmx2.54 mm Pitch			Protectron	P9604-30-15-1
	J3	Box header 2.54 mm, double row straight 20 pin	Header 2x10 pin, 2.54 mmx2.54 mm Pitch			Protectron	P9603-20-15-1
	J4	USB Mini B-Type	USB Mini B-Type			Samtec	MUSB-05-S-B-SM-A
	SW1	RESET switch	Push button			Farnell	9471898
	SW2(DNM)	SPDT switch	Slider			Farnell	674357
LEDs	D1, D2	LED	SMD0805			Any	

Table 4. BOM (continued)

Category	Reference designator	Component Description	Package	Manufacturer	Manufacturer's ordering code / orderable part number	Supplier	Supplier ordering code
Capacitors	C1,C2,C3,C6, C9,C10,C11, C12,C14,C16, C19	100 nF	SMD0805			Any	
	C4,C5	22 pF	SMD0805			Any	
	C15	4.7 µF	SMD1206			Any	
	C7,C18	CAP CER 10 µF 16 V X5R 1206	SMD1206			Digikey	587-1339-2-ND
	C8	10 nF	SMD0805			Any	
	C17	4.7 nF	SMD0805			Any	
Inductors	L1	Inductor multilayer 10 µH 2012	SMD0805			Digi-Key	445-1059-1-ND
Resistors	R1,R5	Res 100 kΩ 1/8 W 5% 0805 SMD	SMD0805			Digi-Key	311-100KARTR-ND
	R3(DNM), R25,R8, R24(DNM), R4	Res 0.0 Ω 1/8 W 0805 SMD	SMD0805			Digi-Key	RMCF1/100RTR-ND
	R6,R14	Res 1 MΩ 1/8 W 5% 0805 SMD	SMD0805			Digi-Key	RMCF1/101MJRTR-ND
	R2,R15	360 Ω	SMD0805			Any	
	R7,R9, R10, R16, R11, R12	10 kΩ	SMD0805			Any	
	R17	100 Ω	SMD0805				
	R13	RES 1.5 kΩ 1/8 W 5% 0805 SMD	SMD0805			Digi-Key	RMCF1/101.5KJRTR-ND

Table 4. BOM (continued)

Category	Reference designator	Component Description	Package	Manufacturer	Manufacturer's ordering code / orderable part number	Supplier	Supplier ordering code
Resistors	R18,R19,R20, R21,R22,R23	4.7 kΩ	SMD0805			Any	
	RN1,RN2, RN4,RN5, RN6,RN7, RN8,RN9	Res array 150 Ω 5% 4 res SMD	1206 (3216 Metric), Convex			Digi-Key	Y9151CT-ND
	RN3	Res array 4.7 kΩ 8TRM 4RES SMD	1207 (3216 Metric), Convex			Digi-Key	YC164J-4.7KCT-ND
Test point	TP1, TP2	Term test point , Slotted .032"DIA	Test points, slotted			Digi-Key	1031K-ND
Scews and nuts	Not applicable	Screw : 02 series: Pan Style 4-40 Screw	Length : 1.00 (25.4) inches (mm) diameter : .210 (5.3) inches (mm)			All electronics hardware	12-02-160
	Not applicable	00 series: Hex Nut	Diameter : .250 (6.4) inches (mm)			All electronics hardware	12-00-440

Appendix B All possible interpretations of the 10-pin interface

Table 5. All possible Interpretations of the 10-pin interface

Pin#	I^2C	UART	SPI	GPIO		ADC	PWM	Supply
				Input pull-up, input floating, input with interrupt falling and rising	Output push-pull and output open drain			
1	SCL	TX	No	Yes	Yes	No	No	No
2	SDA	RX	No	Yes	Yes	No	No	No
3	No	No	No	No	No	No	No	GND
4	No	No	No	No	No	No	No	V_CON
5	No	CTS	MISO	Yes	Yes	No	No	No
6	No	RTS	SCK	Yes	Yes	No	No	No
7	No	No	MOSI	Yes	Yes	No	No	No
8	No	No	NSS	Yes	Yes	No	No	No
9	No	No	No	Yes	Yes	No	Yes	No
10	No	No	No	Yes	Yes	Yes	Yes	No

Appendix C All possible interpretations the of 30-pin interface

Table 6. All possible interpretations of the 30-pin interface

Pin#	I^2C	UART	SPI	GPIO		ADC	PWM	Supply
				Input pull-up, input floating, input with interrupt falling and rising	Output push pull and output open drain			
1	No	No	No	No	No	No	No	5V
2	No	No	No	No	No	No	No	GND
3	No	CTS	No	Yes	Yes	No	No	No
4	No	RTS	No	Yes	Yes	No	No	No
5	No	TX	No	Yes	Yes	No	No	No
6	No	RX	No	Yes	Yes	No	No	No
7	No	No	No	Yes	Yes	No	Yes	No
8	No	No	No	Yes	Yes	No	Yes	No
9	No	No	No	Yes	Yes	No	No	No
10	No	No	No	Yes	Yes	No	No	No
11	SCL	No	No	Yes	Yes	No	No	No
12	SDA	No	No	Yes	Yes	No	No	No
13	No	No	No	No	No	No	No	GND
14	No	No	No	No	No	No	No	V_CON
15	No	No	MISO	Yes	Yes	No	No	No
16	No	No	CLK	Yes	Yes	No	No	No
17	No	No	MOSI	Yes	Yes	No	No	No
18	No	No	NSS	Yes	Yes	No	No	No

Table 6. All possible interpretations of the 30-pin interface (continued)

Pin#	I^2C	UART	SPI	GPIO		ADC	PWM	Supply
				Input pull-up, input floating, input with interrupt falling and rising	Output push pull and output open drain			
19	No	No	No	Yes	Yes	No	Yes	No
20	No	No	No	Yes	Yes	Yes	No	No
21	No	No	No	Yes	Yes	No	Yes	No
22	No	No	No	Yes	Yes	Yes	No	No
23	No	No	No	No	No	No	No	AGND
24	No	No	No	Yes	Yes	Yes	No	No
25	No	No	No	Yes	Yes	Yes	No	No
26	No	No	No	Yes	Yes	No	No	No
27	No	No	No	Yes	Yes	No	No	No
28	No	No	No	No	No	No	No	GND
29	No	TX	No	Yes	Yes	No	No	No
30	No	RX	No	Yes	Yes	No	No	No

Appendix D Tables and figures

Table 7. GPIO modes of 10-pin interface

Mode	Sub-mode	Expected result
Input mode (read)	Input pull-up (default mode)	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 1.
	Input with rising interrupt	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 0. As soon as the value changes from 0 to 1(rising interrupt detected), the interrupt status reads 01 from 00.
	Input with falling interrupt	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 1. As soon as the value changes from 1 to 0(falling interrupt detected), the interrupt status reads 01 from 00.
Output mode (write)	Output push-pull	When the “Write” operation is performed with values 0 or 1, the voltage level on the corresponding GPIO pin can be observed corresponding to the value written.
	Output PWM (only GPIO5 and GPIO6)	Depending on the duty cycle and the frequency settings, the PWM clock is generated.

Figure 34. PWM signal

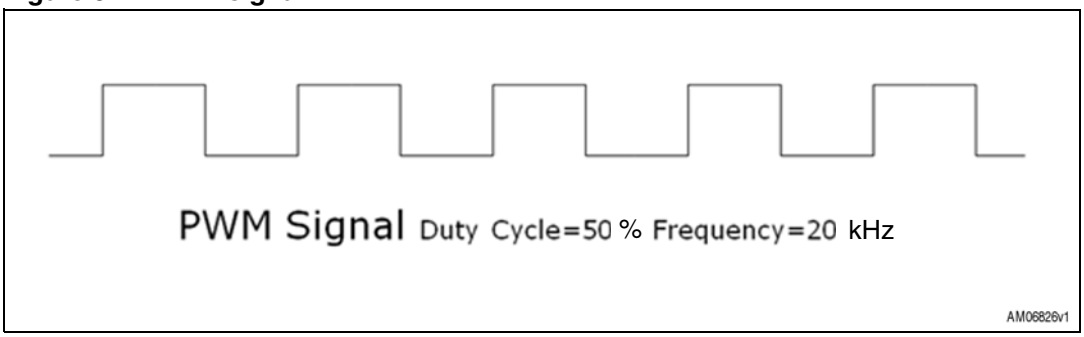


Table 8. GPIO modes of 30-pin interface

Mode	Sub-mode	Expected Result
Input mode (read)	Input pull-up (default mode)	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 1.
	Input floating	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 0 or 1 randomly.
	Input with rising interrupt	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 0. As soon as the value changes from 0 to 1(rising interrupt detected), the interrupt status reads 01 from 00.
	Input with falling Interrupt	When you perform the GPIO “Read” operation, you get the GPIO value as ‘0’ or ‘1’. If no connection is made to this pin, it reads 1. As soon as the value changes from 1 to 0(falling interrupt detected), the interrupt status reads 01 from 00.
Output mode (write)	Output push-pull	When the “Write” operation is performed with values 0 or 1, the voltage level on the corresponding GPIO pin can be observed corresponding to the value written.
	Output open-drain	When the “Write” operation is performed with value 0, the voltage level on the corresponding GPIO pin is 0. When the “Write” operation is performed with value 1, the voltage level on the corresponding GPIO pin can be 0 or 1 randomly.

Table 9. PWM channel settings

PWM channel	Pin # (see *Figure 17*)	Frequency	Duty cycle
PWM channel 1	7	F1	D1
PWM channel 2	8	F1	D2
PWM channel 3	19	F2	D3
PWM channel 4	21	F2	D4

Revision history

Table 10. Document revision history

Date	Revision	Changes
17-Sep-2010	1	Initial release.
21-Sep-2010	2	Typo error in cover page
07-Jan-2011	3	– Modified: *Table 4: BOM* – Modified: title

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com