Atmel EDBG

Embedded Debugger

PRELIMINARY DATASHEET

Description

The Atmel® Embedded Debugger is an intuitive plug-and-play solution which adds full programming and debugging support to embedded hardware kits containing Atmel microcontrollers. It enables seamless integration between the target hardware and Atmel Studio frontend.

In addition, a Virtual COM port provides a UART bridge between the host computer and the target device.

The EDBG is based on the Atmel UC3A4 high-performance low-power 32-bit AVR® microcontroller running at up to 60MHz. The device includes an on-chip USB 2.0 high-speed hardware module with dedicated DMA channels, making it ideal for data communications.

EDBG is available in 100-pin VFBGA package.

EDBG is available pre-programmed with secure bootloader and application image, and can be fully configured over its USB interface, facilitating a fast and simple production environment.

Atmel Studio support is automatic, provided that valid configuration information is provided.

Features

- Enables on-board programming and debugging
  - JTAG, SWD, PDI
- USB Powered
- Virtual COM port (terminal)
- Full Atmel Studio integration
- MCU board and extension board identification system

Table of Contents

1 Introduction

The Atmel Embedded Debugger (EDBG) consists of an Atmel microcontroller device along with associated IP-protected firmware, design guidelines, manufacturing procedures, and other technical details provided in this document.

Designing and manufacturing a product with the EDBG must be done according to the procedures given in this document. Atmel Studio supports EDBG and provides future support through firmware upgrades to the EDBG using its built-in bootloader.

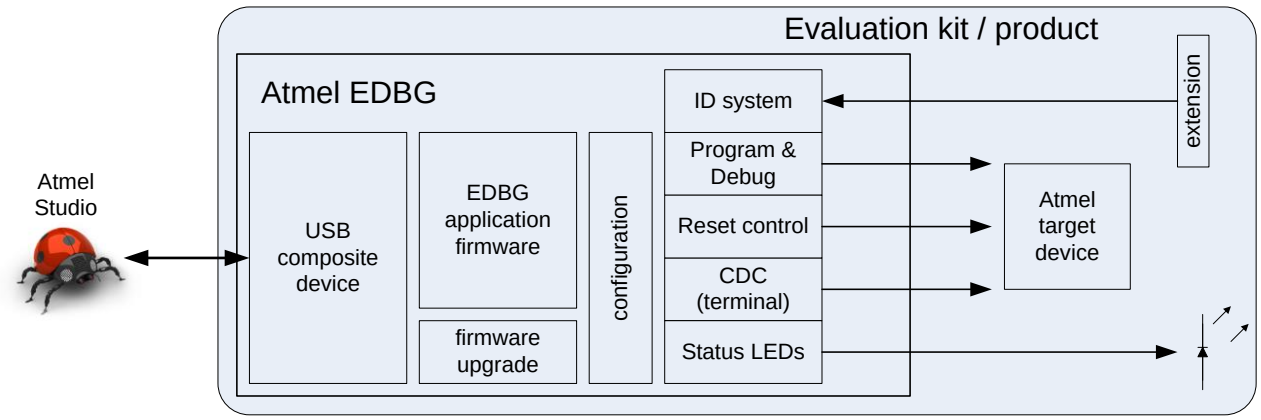
When designing hardware which will include EDBG, each hardware component described is categorized as one of:

- Mandatory EDBG – components which are part of EDBG and cannot be removed
- Required external – components which are required on the board to support EDBG functionality
- Optional feature – EDBG features which can be designed in if applicable

1.1 Overview

A diagrammatic view of the EDBG is shown in Figure 1-1.

Figure 1-1. EDBG Block Diagram



1.2 Abbreviations

ASF: Atmel Software Framework

CDC: Communications Device Class

DW: debugWIRE

EDBG: Embedded Debugger

ISP: In-System Programming

JTAG: Joint Test Action Group

OCD: On-Chip Debug

PCBA: Printed Circuit Board Assembly

PDI: Program and Debug Interface

SWD: Serial Wire Debug

1.3 References

1. Datasheet: AT32UC3A3/A4 Series Complete.

   http://www.atmel.com/devices/AT32UC3A4256.aspx

2. Application Note: AVR32801: 32-bit AVR UC3 A3 series Schematic Checklist.

   http://www.atmel.com/devices/AT32UC3A4256.aspx

3. Software package: Atmel EDBG Development Kit.

   Downloaded from www.atmel.com

   For support, contact: edbg@atmel.com

4. Document: Xplained Pro Extension Design.

   http://gallery.atmel.com/docs/Hardware_Development_Kit.pdf

1.4 Limitations

A. The interfaces listed in Table 2-1 are currently supported. Other interfaces may be supported at a later stage.

B. Only single-supply boards running at 3.3V are currently supported.

C. The EDBG documented here is a subset of the EDBG version used in Atmel Xplained Pro products. (The Data Gateway Interface functionality is not included.) This variant is also referred to as EDBGC.

1.5 Contact Information

For all queries regarding design for/with the Embedded Debugger contact: edbg@atmel.com.

2 Features

2.1 Programming and Debugging

The EDBG can be configured in the factory to provide support for programming and debugging of any Atmel AVR or ARM® microcontroller which includes on-chip debug (OCD) support. A design recommendation is provided in order to correctly connect the target to the EDBG via the appropriate physical interface, and manufacturing parameters in the board configuration are used to inform the EDBG of this connection, which is then used by Atmel Studio front-end.

Available interfaces are shown in Table 2-1.

Table 2-1. Programming and Debugging Interfaces

Interface	Target families	Notes
SWD	Atmel ARM Cortex® processor based MCUs	
JTAG	Atmel ARM Cortex processor based MCUs Atmel AVR UC3 Atmel AVR XMEGA® Atmel megaAVR®	
PDI	Atmel AVR XMEGA	
aWire	Atmel AVR UC3	Not currently available
debugWIRE/ISP	Atmel tinyAVR® and Atmel megaAVR	Not currently available

Consult the datasheet of the target device regarding interface selection.

The appropriate interface will be automatically selected in Atmel Studio for both debug-launch and programming functions. Atmel Studio will not allow selecting another target device or interface, so the EDBG cannot be used as a general purpose debugger.

2.2 Virtual COM Port

The EDBG includes support for a UART bridge to allow the user to send and receive character data from a terminal emulator on the host PC. The EDBG enumerates as a composite device which includes the “EDBG Virtual COM Port” (CDC) device which will appear in the “Ports” section of the Device Manager on the host PC. The device driver is included with Atmel Studio.

Data is channeled through the EDBG and onto the USART COM pins. Consult Section 6.8 for details on how to connect the USART COM pins correctly.

2.2.1 Typical Virtual COM Port Use Case Scenario

The user can achieve printf-style debugging using the Virtual COM Port without making any hardware connections on the board. Either using ASF, or user code, the user can enable the UART transmitter, which is present on the UART pins on the target device.

The user can then open a Terminal window on the host PC, either inside Atmel Studio (after downloading the Terminal extension) or as a standalone application (Putty, HyperTerminal, etc.) A connection must be opened to the device listed as *EDBG Virtual COM Port* with the same baud rate and framing settings as implemented in the firmware on the target application.

Characters written to the UART transmitter on the target will be transported via EDBG to the Terminal window on the host PC. Similarly, data can be sent from the Terminal window to a command handler implemented on the target device.

3 Atmel Studio Support

Correct configuration of the EDBG during manufacturing will lead to seamless Atmel Studio support for the end-user.

3.1 Features

3.1.1 Programming and Debug

Atmel Studio supports programming and debugging of Atmel MCUs with the Embedded Debugger. See Table 2-1 for supported device families and physical interfaces. Each EDBG powered kit is presented with the kit name, and serial number, so that the user easily can distinguish multiple connected kits. If EDBG is correctly configured with target device and physical interface information, Atmel Studio will automatically configure its selection boxes, making the kit easy to use.

3.2 Kit Detection

Upon connection, Atmel Studio detects the name, features, and other parameters of the kit and presents the user with a kit window containing these details. In addition to presenting the parameters stored in the EDBG, the kit window displays a picture of the product, a description text and links to the appropriate documentation. This information comes from an Atmel Studio extension. Contact edbg@atmel.com to make sure your kit is registered with this extension.

4 Board Configuration

4.1 Overview

In order for the EDBG to be aware of the properties and capabilities of the board in which it is mounted, as well as the target device it will operate with, a board configuration is necessary. This information is stored separately to the EDBG firmware, allowing field upgrades to be performed with identical firmware on all deployed kits without any customization required.

The parameters stored in the board configuration include EDBG features, program/debug interface capability, board and manufacturer names and USB serial number.

The board configuration information is stored in flash memory and can be modified freely. However, it is recommended that this information is programmed into the EDBG during the kit manufacturing process and not altered after deployment.

Three blocks of parameters are used by the EDBG.

Table 4-1. Parametric Configuration Blocks

Block	Size	Location	Purpose
Config block	512B	Application Flash memory	Parameters which are typically factory programmed. Must adhere to the memory layout as described in this document.
Behavior block	256B	Flash User Page	Parameters which alter the behavior of the board after deployment to end user. Must adhere to a fixed memory layout.
Custom block	256B	Flash User Page	Parameters which are free for the board manufacturer or end user to use. For example MAC addresses.

4.2 Board Configuration User Interface

When designing and manufacturing a product with EDBG, the engineer must ensure that the EDBG is made aware of the functionality included with that product. Specifying these features inaccurately will compromise the ability of Atmel Studio to be aware of the available features, in turn compromising ease-of-use and in some cases essential functionality.

The EDBG provides a configuration API for both bootloader and debugger modes for this purpose. A set of Python scripts are available for reference.

4.3 Board Configuration Registers (config block)

Table 4-2 lists the available configuration parameters for EDBG.

Table 4-2. Factory Configuration Parameters

Address	ID	Bytes	Name	Parameter	Description
0x00	0x16	4	BCVMR	Board Config Version Minor Register	0x00000001 in this version of this specification.
0x04	0x01	20	BSNR	Board Serial Number Register	Serial number for USB enumeration.
0x18	0x02	50	BNR	Board Name Register	As will be displayed in Atmel Studio. E.g.: “SAM4L Xplained Pro\0” ASCII text, NULL terminated
0x4A	0x03	50	BMR	Board Manufacturer Register	E.g.: “Atmel Corp.\0” ASCII text, NULL terminated

Address	ID	Bytes	Name	Parameter	Description
0x7C	0x04	32	TDNR	Target Device Name Register	Exact device name as used in Atmel Studio. E.g.: “ATxmega128A1U\0” ASCII text, NULL terminated
0x9C	0x08	4	FCR	Feature Configuration Register	Configuration of board features as given in Section 4.3.1
0xA0	0x08	4	PDCR	Program and Debug Capability Register	Configuration of program and debug capability as given in Section 0
0xB0	0x0A	4	EMR	Extension Map Register	Configuration of available extensions as given in Section 4.3.4
0x38	0x17	4	USBR	USB VID/PID Register	Configures the board’s Vendor and Product ID for USB enumeration

4.3.1 Serial Number String

All EDBG implementations must have a unique serial number string. This allows many EDBG devices to be connected to the same USB bus without device conflicts. The serial number will appear in the tool selection boxes in Atmel Studio.

The serial numbers string must adhere to a strict format to facilitate traceability and allow unique identification. The format is given in Table 4-3.

Table 4-3. Serial Number String Format

Address [bytes]	Usage/value	
19, 18, 17, 16	Vendor code (4 bytes): “ATML”	
15, 14, 13, 12	Identifier (4 bytes): “EDBG”	
11, 10,	Version (2 bytes): “C0”	
9, 8,	Serial (sequence) number (10 bytes)	
7, 6, 5, 4		
3, 2, 1, 0		

The fields listed in Table 4-3 are to be partially customized by the vendor:

- Vendor code is a 4-character ASCII tag used to identify the vendor as part of the serial number. (A full string is located in the factory config.) Atmel allocates the vendor code to partners, at which point it is included as part of the manufacturing software provided by Atmel, and thus cannot be changed.
- Identifier is always “EDBG”
- Version: The current version described in this document is “C0”
- The 10-digit serial (sequence) number should be generated by the factory, and be unique (optionally sequential). Atmel make use of the sequential barcodes placed on each PCBA to allocate serial numbers which are thus traceable from the USB enumeration data back to PCB assembly in the factory.

The SERIAL NUMBER SELECT pin can also be used to force the EDBG to use pseudo-random serial numbers, as described in Section 6.4.5.

4.3.2 FCR - Feature Configuration Register

The Feature Configuration Register is programmed according to the features available on the board during production.

Bits:

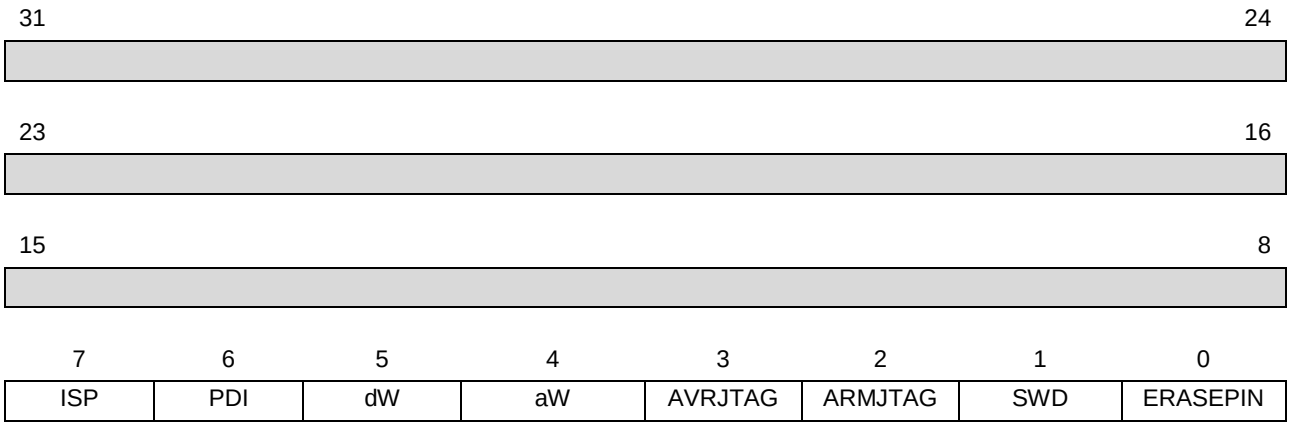
31	30	29	28	27	26	25	24
FUR							
23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8
				TVR	USBVCOM		
7	6	5	4	3	2	1	0

- **Bit 31-12: Reserved**
- **Bit 9-0: Reserved**
- **FUR: Firmware Upgrade Rejection**
  1: Automatic firmware upgrades are not possible.
  0: Firmware can be upgraded automatically.
- **TVR: Target Voltage Read**
  1: Target voltage can be read by Atmel Studio.
  0: Target voltage cannot be read by Atmel Studio.
- **USBVCOM: USB Virtual COM port**
  1: USB virtual COM port available on the board.
  0: USB virtual COM port not available on the board.

4.3.3 PDCR – Program and Debug Capability Register

The Program and Debug Capability Register is used to store the program and debug capabilities that are supported by a specific board.

Bits:



- **Bit 31-8: Reserved**
- **Bit 11-8: Reserved**
- **ISP: Program and Debug Mode ISP**
  1: Program and Debug Mode ISP supported by the board.
  0: Program and Debug Mode ISP not supported by the board.
- **PDI: Program and Debug Mode PDI**
  1: Program and Debug Mode PDI supported by the board.
  0: Program and Debug Mode PDI not supported by the board.
- **dW: Program and Debug Mode debugWIRE**
  1: Program and Debug Mode debugWIRE supported by the board.
  0: Program and Debug Mode dW not supported by the board.
- **aW: Program and Debug Mode aWire**
  1: Program and Debug Mode aWire supported by the board.
  0: Program and Debug Mode aWire not supported by the board.
- **AVRJTAG: Program and Debug Mode AVR JTAG**
  1: Program and Debug Mode AVR JTAG supported by the board.
  0: Program and Debug Mode AVR JTAG not supported by the board.
- **ARMJTAG: Program and Debug Mode ARM JTAG**
  1: Program and Debug Mode ARM JTAG supported by the board.
  0: Program and Debug Mode ARM JTAG not supported by the board.
- **SWD: Program and Debug Mode Serial Wire Debug**
  1: Program and Debug Mode SWD supported by the board.
  0: Program and Debug Mode SWD not supported by the board.
- **ERASEPIN: Chip Erase Pin Available**
  Some SAM devices have a dedicated ERASE pin. To avoid permanently locking such devices, it is recommended to allow the user to have access to this pin to recover the device. Set this bit if the ERASE pin is to be controlled by the EDBG.
  1: Chip Erase Pin is connected and available on this board.
  0: Chip Erase Pin is not available on this board.

4.3.4 EMR – Extension Map Register

The EMR register indicates which extension headers are available on the board. This information is used to identify which ID lines that need to be read. For more information, see reference (4).

Bits:

31	30	29	28	27	26	25	24

23	22	21	20	19	18	17	16

15	14	13	12	11	10	9	8

7	6	5	4	3	2	1	0
	EXT7	EXT6	EXT5	EXT4	EXT3	EXT2	EXT1

- **Bit 31-7: Reserved**
- **EXTn:**
  1: EXTn available.
  0: EXTn not available on the board.

4.3.5 USBR – USB VID/PID Register

The USBR configures the board's USB Vendor ID and Product ID, which are used during enumeration on the USB bus to identify the board. A USB Vendor ID can be purchased from usb.org. Product ID is freely configurable. If the ATMEL Vendor ID is used, the only the EDBGC product ID is allowed.

Bits:

31	30	29	28	27	26	25	24
VID[15:8]							

23	22	21	20	19	18	17	16
VID[7:0]							

15	14	13	12	11	10	9	8
PID[15:8]							

7	6	5	4	3	2	1	0
PID[7:0]							

- **Bit 31-16:**
  Vendor ID.
  Purchased from usb.org. Atmel Vendor ID can be used (0x03EB).

- **Bit 15-0:**
  Product ID.
  Fully configurable. If Atmel Vendor ID is used, then this field defaults to 0x216A.

5 Package and Pinout

The EDBG is available in a 100-pin VFBGA package only.

All pins marked as "-" in Figure 5-1 must be treated as Reserved and left not connected to ensure backwards compatibility. Future features and modes may make use of these pins.

Pin functionality is listed in Table 5-1.

Note: Some pins are duplicated in this table since they have different functionality with different target devices.

Figure 5-1. VBGA Pinout

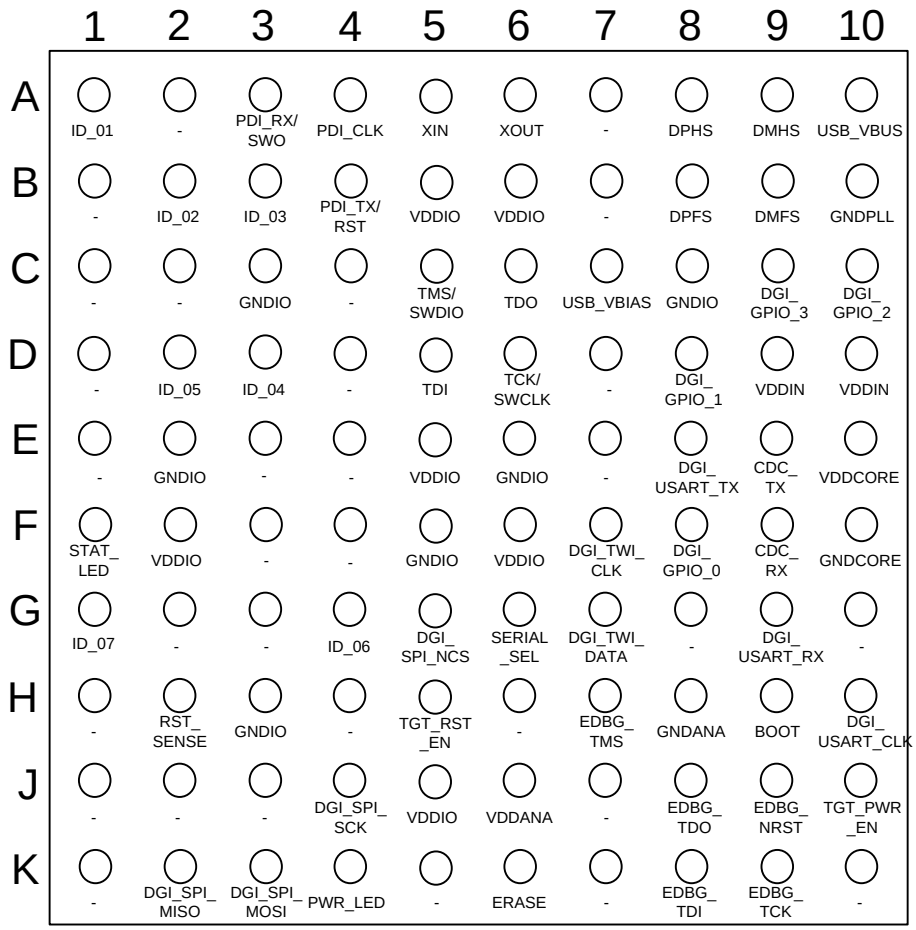


Table 5-1. Pin Descriptions

BGA	EDBG PIN	Function	Description
General EDBG pins			
F1	STAT_LED	STATUS LED	EDBG status LED. Active low. FET driver recommended. See Section 6.4.3.
K4	PWR_LED	POWER LED	EDBG power LED. Active low. FET driver recommended. See Section 6.4.3.
H9	BOOT	FORCE BOOT	EDBG boot mode emergency recovery pin. Test-point close to GND to allow user to recover EDBG board. See Section 6.4.4.

BGA	EDBG PIN	Function	Description
G6	SERIAL_SEL	SERIAL SELECT	Selects mode for USB serial number generation. See Section 6.4.5 for more information.
J10	TGT_PWR_EN	TARGET POWER ENABLE	Enable/disable target power
EDBG manufacturing programming pins. All pins should be available during the manufacturing procedure. See Section 6.4.1.1.			
K9	EDBG_TCK	TCK	JTAG Test Clock pin for EDBG device
H7	EDBG_TMS	TMS	JTAG Test Mode Select pin for EDBG device
K8	EDBG_TDI	TDI	JTAG Test Data In for EDBG device
J8	EDBG_TDO	TDO	JTAG Test Data Out for EDBG device
J9	EDBG_NRST	/RESET	/RESET for EDBG device
Extension identification system pins			
A1	ID_01	ID 01	Extension identification system channel 1
B2	ID_02	ID 02	Extension identification system channel 2
B3	ID_03	ID 03	Extension identification system channel 3
D3	ID_04	ID 04	Extension identification system channel 4
D2	ID_05	ID 05	Extension identification system channel 5
G4	ID_06	ID 06	Extension identification system channel 6
G1	ID_07	ID 07	Extension identification system channel 7
CDC pins			
F9	COM_RX	COM RX	Virtual COM Port receive data
E9	COM_TX	COM TX	Virtual COM Port transmit data
General target interface pins			
H2	RST_SENSE	RESET SENSE	Sense pin for target reset events
B4	PDI_TX/RST	TARGET RESET	Target device reset drive
H5	TGT_RST_EN	TARGET RESET ENABLE	Enables a reset button on the target reset line when this line is also used for communication
JTAG target interface pins			
D6	TCK/SWCLK	TCK	Target device JTAG Test Clock
C5	TMS/SWDIO	TMS	Target device JTAG Test Mode Select
D5	TDI	TDI	Target device JTAG Test Data In
C6	TDO	TDO	Target device JTAG Test Data Out
B4	PDI_TX/RST	TARGET RESET	Target device reset drive
K6	ERASE	ERASE	Target device erase pin (SAM only)
SWD target interface pins			
A3	PDI_RX/SWO	TRACESWO	Target device Serial Wire Output
C5	TMS/SWDIO	SWDIO	Target device Serial Wire Data In/Out
D6	TCK/SWCLK	SWCLK	Target device Serial Wire Clock
K6	ERASE	ERASE	Target device erase pin (SAM only)

BGA	EDBG PIN	Function	Description
PDI target interface pins			
A4	PDI_CLK	PDI CLOCK	Target device PDI clock
A3	PDI_RX/SWO	PDI DATA RX	Target device PDI data receive
B4	PDI_TX/RST	PDI DATA TX	Target device PDI data transmit
EDBG USB pins			
A8	DPHS	DPHS	USB High-Speed Data +
A9	DMHS	DMHS	USB High-Speed Data -
B8	DPFS	DPFS	USB Full-Speed Data +
B9	DMFS	DMFS	USB Full-Speed Data -
A10	USB_VBUS	USB_VBUS	USB VBUS signal
C7	USB_VBIAS	USB_VBIAS	USB VBIAS reference
EDBG power, ground, and clock pins			
A5	XIN	XTAL IN	Crystal Input
A6	XOUT	XTAL OUT	Crystal Output
B5, B6, E5, F2, F6, J5	VDDIO	VDDIO	I/O power supply
D9, D10	VDDIN	VDDIN	Voltage regulator input supply
J6	VDDANA	VDDANA	Analog power supply
E10	VDDCORE	VDDCORE	Voltage regulator output for digital supply
B10	GNDPLL	GNDPLL	PLL ground
C3, C8, E2, E6, F5, H3	GNDIO	GNDIO	I/O ground
H8	GNDANA	GNDANA	Analog ground
F10	GNDCORE	GNDCORE	Digital ground

6 Board Layout and Routing

6.1 Schematic Symbols and Part Numbers

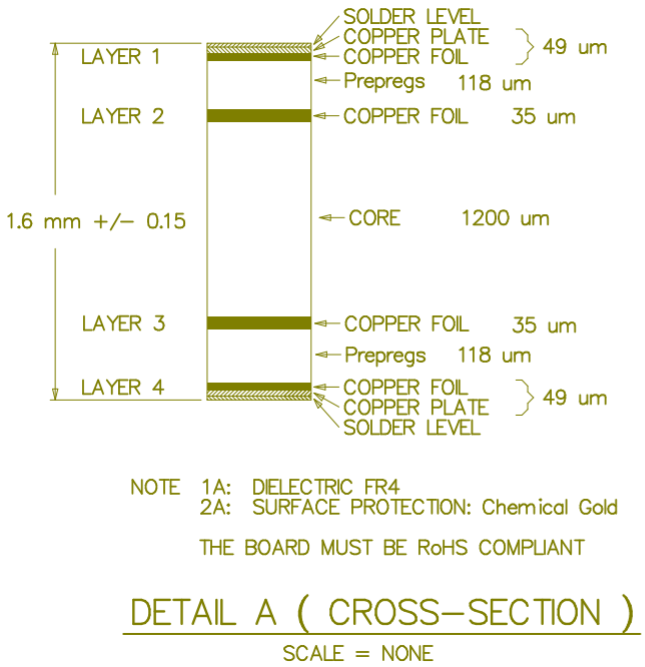
See reference (3) for information on downloading the EDBG Development Kit. The kit contains schematic snippets, bill-of-materials with recommended part numbers, and other useful design resources.

6.2 PCB Stack-up

There are no explicit requirements for PCB stack-up on boards designed with EDBG. To optimally route the required signals from the VFBGA100 package it is recommended to use a board stack of four or more layers.

An example is shown in Figure 6-1.

Figure 6-1. Recommended 4-layer PCB Stack-up



6.3 Power Supply Requirements

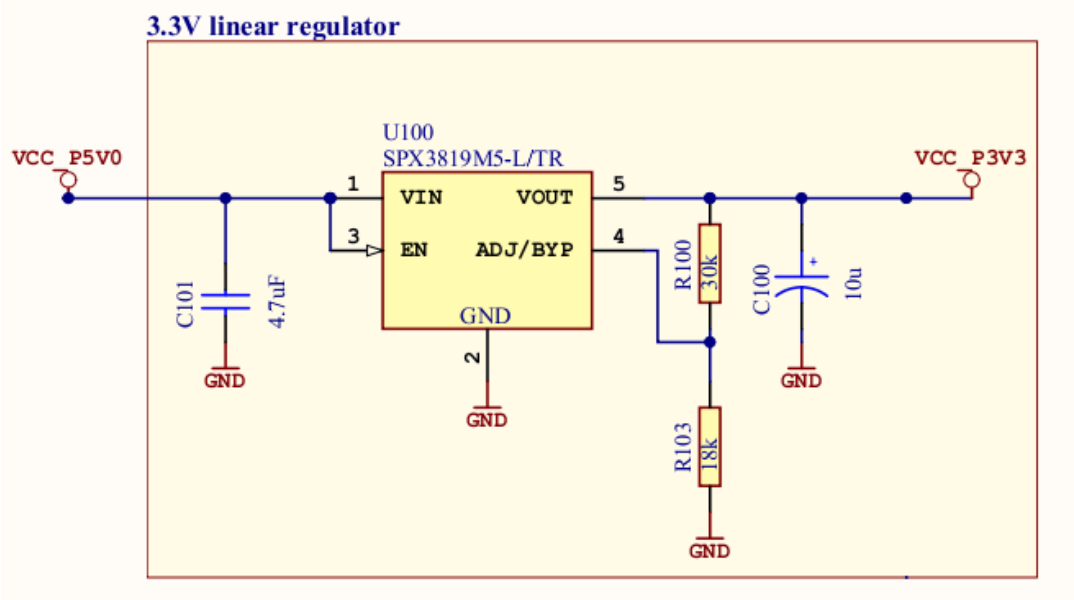
The EDBG design does not include a power supply. This means that the EDBG and associated circuitry must be integrated into the board in which it is to operate. The EDBG operates off a single-supply 3.3V power supply, which must be regulated to provide less than ±50mV ripple at maximum current draw.

The EDBG draws about 100mA when operating at high-speed with all features enabled and in use. The power supply must be designed to provide this with good margin, in addition to powering LEDs and other external circuitry.

When powered by the USB host, the enumeration will request up a maximum of 500mA. This is sufficient to comfortably supply power to the EDBG and other external circuitry on the board with little likelihood of violating the USB power quota under normal operation. For USB-powered boards, the power supply used in Atmel Xplained Pro boards can be used as reference. If the power budget of the board exceeds 500mA, an external power source is required.

An example power supply (as used in Xplained Pro) is shown in Figure 6-2.

Figure 6-2. Example 3.3V Power Supply Schematic Section



6.4 General EDBG Circuitry

6.4.1 EDBG MCU

The EDBG is based on the AT32UC3A4256S 32-bit AVR microcontroller in BGA100 package. A schematic diagram is provided (consult reference (3)) which can be pruned according to individual board requirements.

6.4.1.1 EDBG Factory Programming

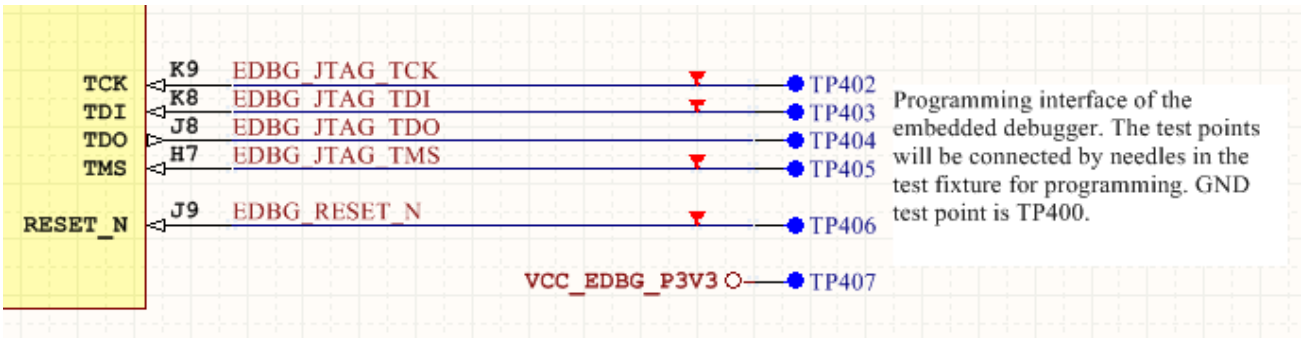
EDBG devices are shipped pre-programmed with both bootloader and application binaries. This means that the JTAG connection described here is optional, although it could be useful for enhanced debugging or troubleshooting, especially on early prototypes.

The EDBG can be programmed using the JTAG pins given in Table 6-1. These can be exposed by pin header or test points according to preference of the manufacturer.

Table 6-1. EDBG Manufacturing Programming Pins (optional)

EDBG pin	EDBG pin name	Signal	Function
K9	EDBG_TCK	TCK	JTAG Test Clock pin for EDBG device
H7	EDBG_TMS	TMS	JTAG Test Mode Select pin for EDBG device
K8	EDBG_TDI	TDI	JTAG Test Data In for EDBG device
J8	EDBG_TDO	TDO	JTAG Test Data Out for EDBG device
J9	EDBG_NRST	/RESET	/RESET for EDBG device

Figure 6-3. EDBG Manufacturing Programming Connections Schematic Section



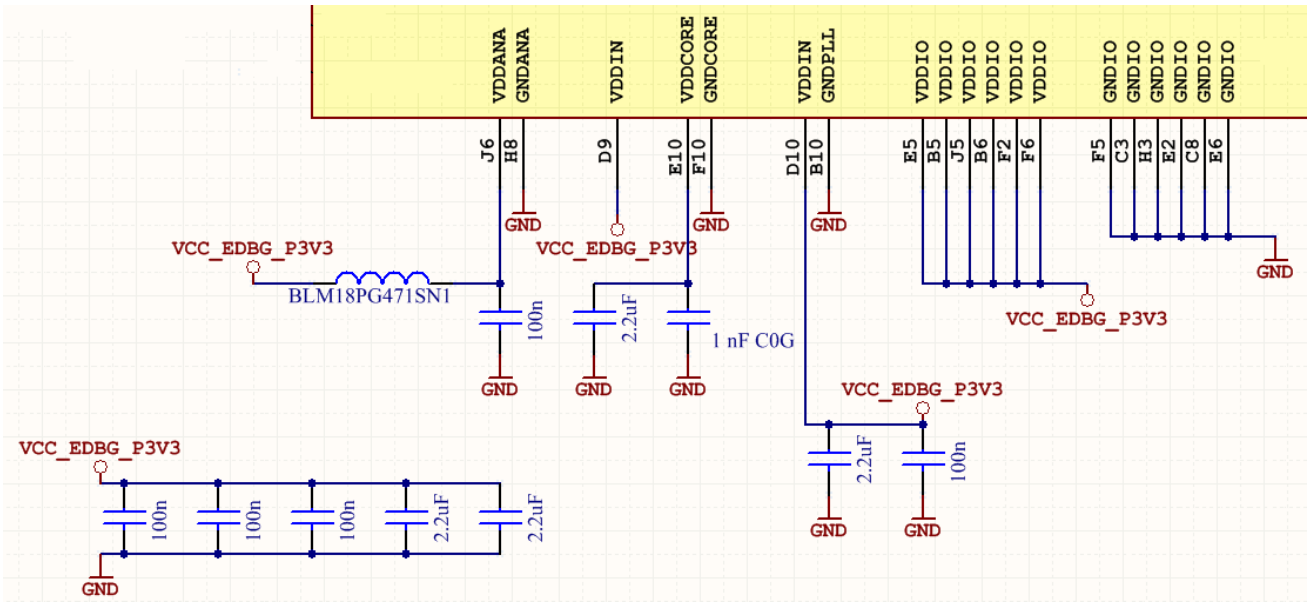
6.4.1.2 Power Supply and Decoupling

EDBG is based on a high-performance 32-bit AVR Microcontroller, which needs a reliable power supply and adequate decoupling to operate correctly. The MCU embeds a voltage regulator that converts from 3.3V to 1.8V from which the core operates. Adequate output supply decoupling is mandatory for VDDCORE to reduce ripple and avoid oscillations. Adequate input supply decoupling is mandatory for VDDIN in order to improve startup stability and reduce source voltage drop. Even if the ADC functionality on the EDBG is not used, it is recommended to include a decoupling / filtering capacitor on VDDANA to improve startup stability.

Table 6-2. Power Supply and Decoupling Requirements

Function	Pins	Recommendation
Analog supply	VDDANA, GNDANA	Power VDDANA from 3.3V through a BLM filter. Decouple with 100nF.
Core regulator output	VDDCORE, GNDCORE	Decouple with 2.2µF and 1nF. Do *not* connect to 3.3V.
Voltage regulator input	VDDIN	Decouple with 2.2µF and 100nF
I/O supply	VDDIO	Decouple with at least 2 × 2.2µF and 3 × 100nF. Distribute decoupling capacitors.

Figure 6-4. Power Supply and Decoupling Schematic Section



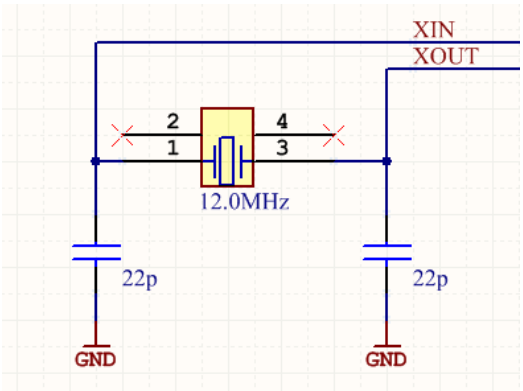
6.4.1.3 Crystal

EDBG requires a 12MHz crystal connected to XIN and XOUT pins for correct operation of its high-speed USB interface.

Atmel uses *Fox FQ5032B 12.0MHz SMD crystal 738B-12* with 22pF load capacitors in its own designs using EDBG.

For electrical requirements of the crystal, see Chapter 9.

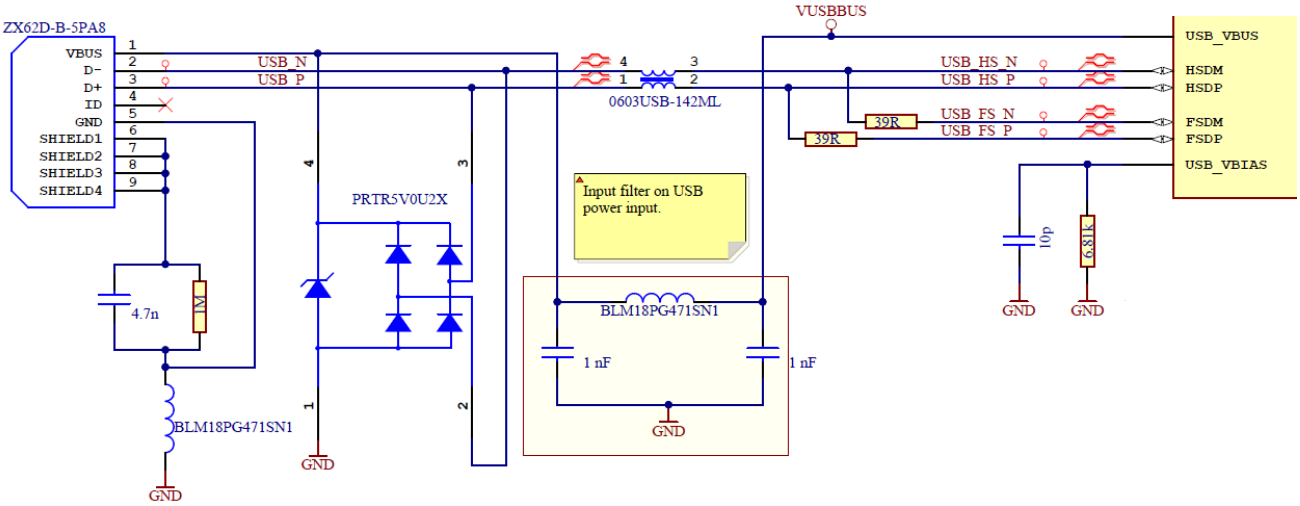
Figure 6-5. Crystal Schematic Section



6.4.2 USB

The EDBG MCU includes a USB 2.0 high-speed controller which requires careful attention during layout and routing. A schematic section is shown in Figure 6-6.

Figure 6-6. USB Schematic Section



The board designer can choose to make use of USB micro, mini, or standard connector. Protection diodes are mandatory on data and power lines. If the VBUS voltage will be regulated and used as power supply, the filter should be in place. USB_VBIAS pin must be connected to ground through a 6810Ω 1% resistor in parallel with a 10pF capacitor.

A common-mode choke is included in the USB data path on some designs. This component is optional, but recommended for EMC purposes.

6.4.2.1 USB High-speed Routing Requirements

Special care must be taken when routing the USB signals to the EDBG. The following rules are recommended:

A. Controlled impedance for USB tracks on top layer with 45Ω single-ended and 90Ω differential.

B. USB routing is to be done with 0.18mm track width and 0.22mm spacing.

Note: The recommendations here are only valid for the PCB stack-up as described in Section 6.2. It is advisable to calculate these parameters according to the PCB stack-up that will be used.

Reference (2) contains further details as well as a layout and routing recommendation for the USB subsystem.

6.4.3 LEDs

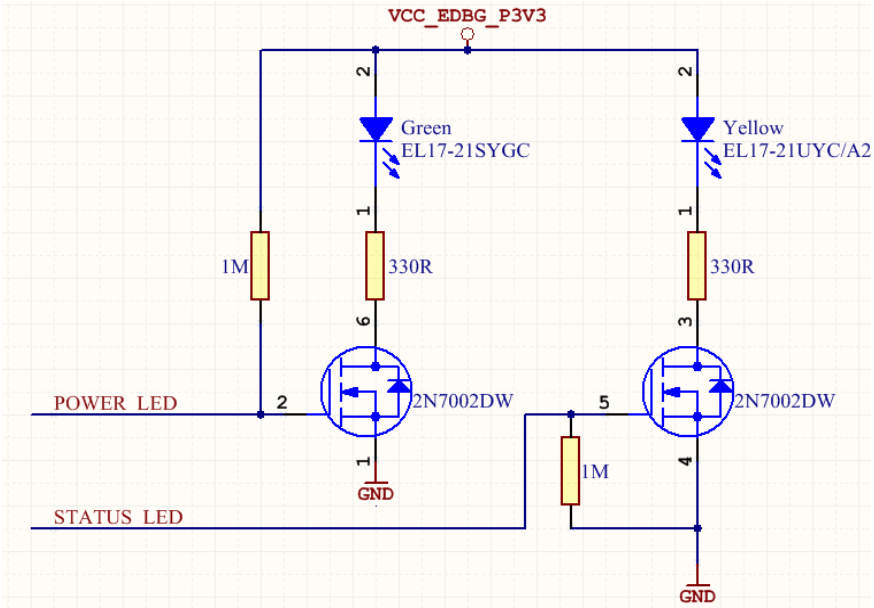
The EDBG controls two LEDs to indicate its status. To provide a uniform behavior with other kits, both LEDs should be included with the colors and default states indicated in Table 6-3.

Table 6-3. LED Requirements

Function	Control	Color	Default state	Usage
Power	PWR_LED	Green	ON	Indicates presence of power to the EDBG
Status	STAT_LED	Yellow	OFF	Indicates status of the EDBG

Due to limited drive strength of the I/O pins on the EDBG MCU, a FET driver is recommended. Pull-ups are used to switch the FET on or off in its default state before the EDBG actively drives the control signal.

Figure 6-7. LED Schematic Section



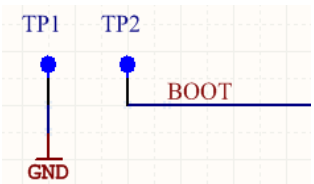
6.4.4 BOOT Pin

During normal operation, firmware upgrades to EDBG are initiated by Atmel Studio sending a command to the EDBG requesting it to reboot in upgrade mode. In the unlikely event of corruption in the application image preventing USB enumeration, the end-user has the ability to force the EDBG into its upgrade mode. This mode is programmed and locked during manufacture, and is always available.

To force boot into upgrade mode, the user should pull the BOOT pin to ground while powering up the EDBG.

Since this is not regarded as normal operation, it is recommended that the BOOT function be made available using pin header, test points, or not-mount resistor footprint.

Figure 6-8. BOOT Pin Schematic Section



6.4.5 SERIAL NUMBER SELECT Pin

This pin is used to select between two modes of USB serial-number generation in un-configured devices. When EDBG is powered up, its USB serial number is read from the board configuration during enumeration. If the string is invalid, or un-configured, the SERIAL NUMBER SELECT pin is sampled and one of the following actions taken:

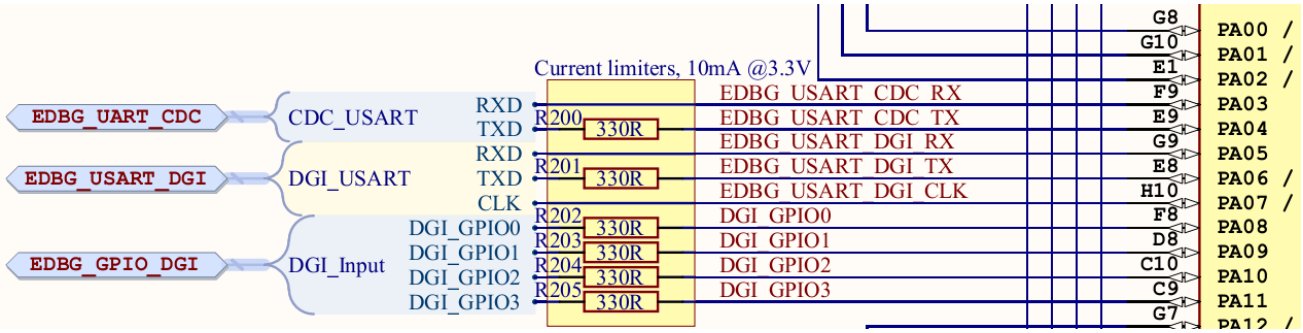
- Floating (internally pulled up): The serial number string is set to “ATMLEDBGC0FFFFFFFFFF”. During manufacturing a connection can be made to this device, and serial number set to a custom value stored in the board configuration, which will then come into effect after the next EDBG reset.
- Tied low: The serial number is set to “ATMLEDBGC0” + 10 characters based on a hash of factory/wafer parameters. This is not guaranteed to be unique, but has enough randomness for it to be used as a unique serial number for end users.

6.4.6 Short Circuit Protection

All I/O lines that can be driven by the EDBG and at the same time by the target MCU are potential short circuits. To avoid this, a current limiting resistor is placed in series on all affected I/O lines.

I/O drive capability is limited by the capability of the internal power rails related to driving or sinking I/O voltage. The maximum output is about 60mA per power-pair and a typical pad can drive/sink up to about 20mA at 25°C without damage. To add a fair margin, it is recommended to set the EDBG short circuit protection to 10mA per I/O pin (330Ω series resistor).

Figure 6-9. I/O Short Circuit Protection Schematic Section



6.5 Extensions and Extension Identification

The EDBG supports a system for identifying Xplained Pro compatible extension boards which are connected to the system. The Extension boards are peripheral extensions to the *target* device, but their identification system is connected to the EDBG to enable Atmel Studio to detect which extensions are attached. To make use of extension headers in the product, the designer must:

- Place a fully mechanically and electrically compatible extension header on the board
- Route the extension header to the target device according to the published extension pin-out
- Route the ID line to one of the EDBG EXTID<n> channels
- Enable the corresponding bit in the factory configuration to indicate to the EDBG that EXTID<n> is connected

For more information on Xplained Pro compatible extensions, consult reference (4).

6.6 Target Programming Interfaces

6.6.1 JTAG

Target devices which support JTAG interface are connected by four direct connections on each of the JTAG lines as shown in Table 6-4. In addition, the target reset line must be controlled by the EDBG, since it is essential in debug launch scenarios. To protect the EDBG in cases where user application code overrides the JTAG pins on the target during a debug session, current limiting resistors (330Ω) should be in place on TCK, TMS, and TDI lines.

Some ARM Cortex processor based MCU devices have a dedicated ERASE pin – this is essential in certain debug launch scenarios, and must be routed.

Table 6-4. JTAG Target Connections

EDBG pin	EDBG signal	JTAG signal	Description	Target
D6	TCK/SWCLK	TCK	Target device JTAG Test Clock	Connect to TCK
C5	TMS/SWDIO	TMS	Target device JTAG Test Mode Select	Connect to TMS
D5	TDI	TDI	Target device JTAG Test Data In	Connect to TDI
C6	TDO	TDO	Target device JTAG Test Data Out	Connect to TDO
B4	PDI_TX/RST	TARGET RESET	Target device reset drive	Connect via reset circuitry
K6	ERASE	ERASE	Target device erase pin (SAM only)	Connect to ERASE

6.6.2 SWD

Target devices which support SWD interface are connected by two or three direct connections, as shown in Table 6-5. In addition, the target reset line must be controlled by the EDBG, since it is essential in debug launch scenarios.

Some ARM Cortex processor based MCU devices also have a dedicated ERASE pin which is essential in certain startup scenarios.

The TRACESWO pin is optional, although it is recommended to be connected. The TRACESWO pin can be used by on-chip modules to send serial trace data to the EDBG while in run mode, allowing for example printf output.

To protect the EDBG in cases where user application code overrides the SWD pins on the target during a debug session, current limiting resistors (330Ω) should be in place on SWCLK and SWDIO lines.

Table 6-5. SWD Target Connections

EDBG pin	EDBG signal	SWD signal	Description	Target
D6	TCK/SWCLK	SWCLK	Target device Serial Wire Clock	TCK/SWCLK
C5	TMS/SWDIO	SWDIO	Target device Serial Wire Data In/Out	TMS/SWDIO
B4	PDI_TX/RST	TARGET RESET	Target device reset drive	Connect via reset circuitry
A3	PDI_RX/SWO	TRACESWO	Target device Serial Wire Output	TDO/TRACESWO
K6	ERASE	ERASE	Target device erase pin	ERASE

6.6.3 PDI

Target devices which support PDI interface have to be connected to EDBG using external components. The required pins are shown in Table 6-6.

Table 6-6. PDI Target Connections

EDBG pin	EDBG signal	PDI signal	Description
A4	PDI_CLK	PDI CLOCK	Target device PDI clock
A3	PDI_RX/SWO	PDI DATA RX	Target device PDI data receive
B4	PDI_TX/RST	PDI DATA TX	Target device PDI data transmit

6.6.3.1 PDI Clock

PDI_CLK must be connected directly between EDBG and the target's RESET pin.

Because the RESET line is shared with the PDI clock, special handling is required if a manual reset push-button switch is required on this line in addition to EDBG control. See Section Table 6-7 for more details.

6.6.3.2 PDI Data

PDI_DATA is a bi-directional signal, which must be multiplexed into two component channels for transmission and reception by a USART module on the EDBG. This is done by simply tying the PDI_TX and PDI_RX lines together and connecting this signal to the PDI_DATA line. EDBG will tristate PDI_TX during reception.

6.6.4 aWire

This interface is currently not available on EDBG. Contact Atmel Technical Support if you are interested in implementing this.

6.6.5 debugWIRE / ISP

This interface is currently not available on EDBG. Contact Atmel Technical Support if you are interested in implementing this.

6.7 RESET Line Considerations

6.7.1 Reset Control for Devices with Communication on Target Reset Line (PDI)

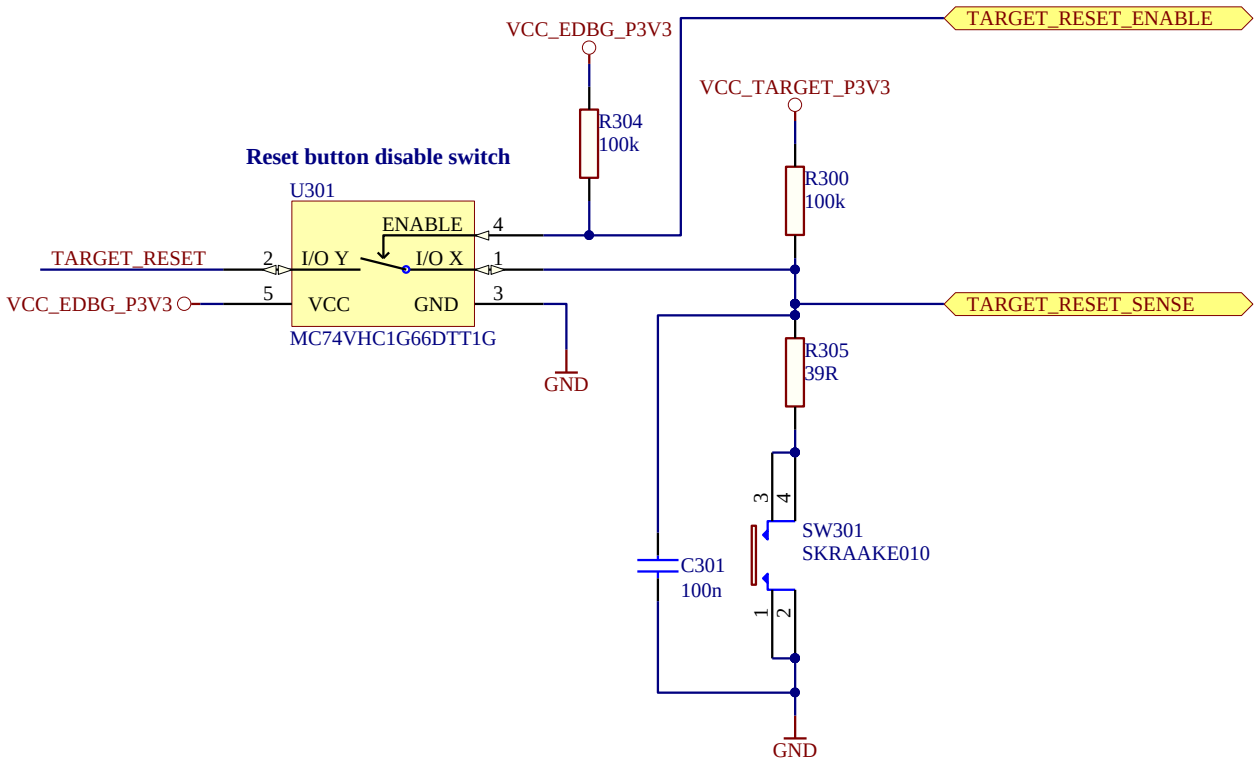
In this configuration it is possible to disable the reset button so that communication on the reset line cannot be corrupted by activity on the reset button. In addition it is possible to detect if the button is pushed while communication is ongoing on the reset line. This enables the EDBG firmware to handle this case gracefully by interrupting ongoing communications in a safe way, resetting the target actively (internal reset) and reporting back to Atmel Studio that a target reset was performed by the user during ongoing communication.

Signals required for this implementation are shown in Table 6-7.

Table 6-7. Shared Reset Line Requirements

Signal	Function
RST_SENSE	This signal senses the state of the reset button
TARGET_RESET	This signal drives or pulls the reset line during data transfer from the EDBG to the target
TGT_RST_EN	By default the reset button is enabled but when the EDBG starts to communicate over the reset line it must disable the buttons capability to pull the reset line low. The enabling and disabling of the pull-down function of the reset button is controlled with this signal.

Figure 6-10. Target Reset Shared Configuration



The additional resistor that is connected directly to the button reduces the risk that the reset signal will oscillate due to the high peak current once the button is pushed. The peak current depends on the charged capacitance that is present on the reset line, it is mainly the filter capacitor but also track and pin capacitances. Directly discharging these capacitances to ground will lead to a high initial peak current, which will then oscillate due to parasitic inductance. This oscillation will lead to negative voltages on reset and can possibly destroy the device.

Care must be taken that the V_{IL} level of the target MCU is reached when the button is pushed since the switch will introduce an additional resistance, in this case (MC74VHC1G66) up to 100Ω. This resistance plus the current limiter resistance must be considered in relation to the pull-up that is used for the communication via the reset line.

$$V_{IL} > VCC\_TARGET\_P3V3 \frac{R_{switch} + R_{limit}}{R_{pull-up} + R_{switch} + R_{limit}}$$

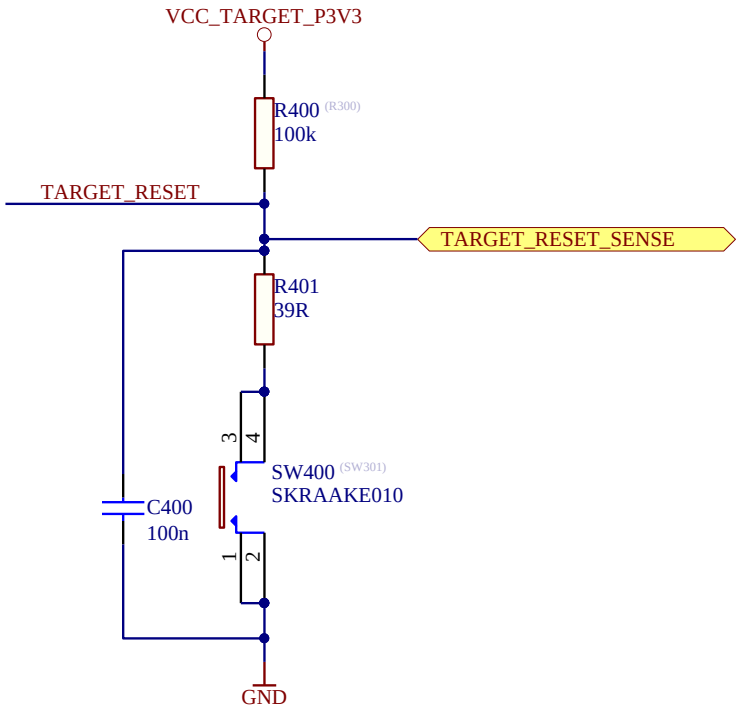
6.7.2 Normal Reset Control

When no communication via the reset signal is required the circuitry for the reset button is much simpler. Two signals from the EDBG are required, as shown in Table 6-8.

Table 6-8. Normal Reset Controls Requirements

Signal	Function
TARGET_RESET	EDBG control of the reset. The EDBG will pull this signal low to enable reset. This signal is directly connected to the reset of the target MCU.
RST_SENSE	Sense signal for the EDBG to detect the actual state of the reset line. This signal is actually not required for this configuration but to keep the EDBG firmware the same for all implementations this signal must be added.
TGT_RST_EN	To be left not-connected in this mode

Figure 6-11. Target Reset Normal Configuration



6.8 Virtual COM Port Interface

The Virtual COM Port interface provides a bridge between the target device's UART module and a Virtual COM Port enumerated on the host PC. Connections required to support the Virtual COM Port functionality are shown in Table 6-9.

Table 6-9. Virtual COM Port Connectivity Requirements

EDBG pin	EDBG pin name	Target pin	Function
F9	COM_RX	UART TXD	Data from target to EDBG
E9	COM_TX	UART RXD	Data from EDBG to target

Note: Virtual COM Port must be enabled in the factory configuration!

Note: It is advised to include a current limiting 330Ω resistor on the COM_TX path to reduce the effect of contention should the target application drive this pin unintentionally.

USART is not a bus! Connect COM_RX to TXD and COM_TX to RXD!

7 Board Manufacturing

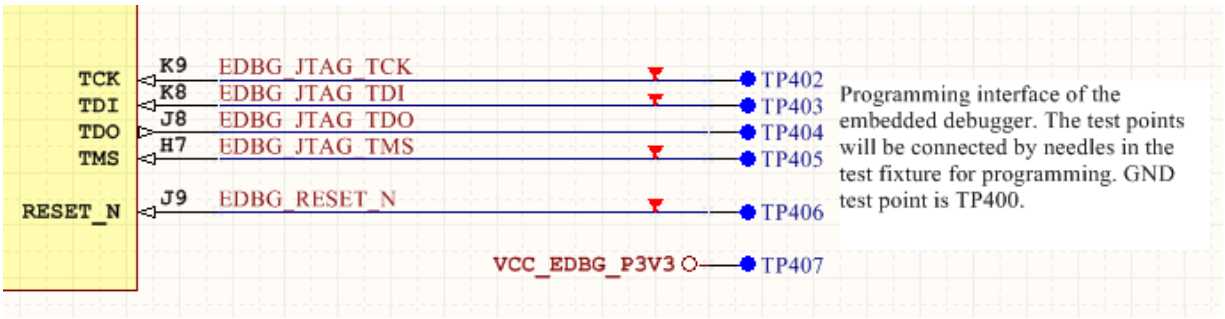
7.1 EDBG Hardware Connections

EDBG devices are shipped pre-programmed with both bootloader and application binaries. This means that the JTAG connection described here is optional, although it could be useful for enhanced debugging or troubleshooting, especially on early prototypes.

Table 7-1. EDBG Manufacturing Programming Pins (optional)

EDBG pin	EDBG pin name	Signal	Function
K9	EDBG_TCK	TCK	JTAG Test Clock pin for EDBG device
H7	EDBG_TMS	TMS	JTAG Test Mode Select pin for EDBG device
K8	EDBG_TDI	TDI	JTAG Test Data In for EDBG device
J8	EDBG_TDO	TDO	JTAG Test Data Out for EDBG device
J9	EDBG_NRST	/RESET	/RESET for EDBG device

Figure 7-1. EDBG Manufacturing Programming Connections Schematic Section



8 Ordering Information

8.1 Atmel EDBG Ordering Codes

Atmel ordering code	Package type	Voltage range	Temperature range
AT32UC3A4256HHB-C1UR	100-ball VFBGA	3.3V	Industrial (-40ºC to +85ºC)

9 Electrical Characteristics

For characteristics not listed here, consult the relevant section of the datasheet reference (1).

9.1 Absolute Maximum Ratings

Operating Temperature	-40°C to +85°C
Storage Temperature	-60°C to +150°C
Voltage on Input Pin with respect to Ground	-0.3V to 3.6V
Maximum Operating Voltage (VDDCORE)	1.95V
Maximum Operating Voltage (VDDIO)	3.6V
Total DC Output Current on all I/O Pin	
for TQFP144 package	370 mA
for TFBGA144 package	370 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

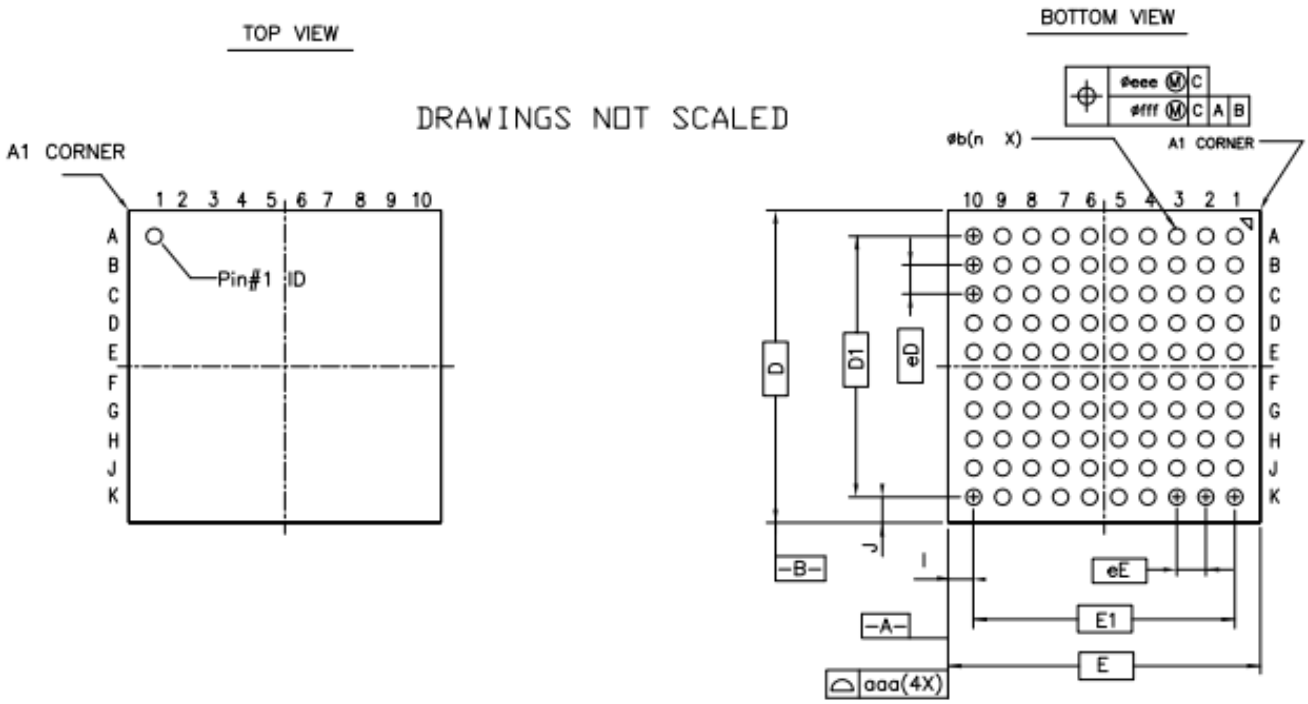
10 Mechanical Characteristics

10.1 Thermal Considerations

Consult the datasheet reference (1).

10.2 Package Drawings

VFBGA-100 package drawing.



	MM		
	MIN	NOM	MAX
A	----	----	1.000
A1	0.220	----	0.320
M	0.450 BSC		
S	0.210 BSC		
b	0.300	----	0.400
E/D	7.00 +/- 0.100		
e	0.65 BSC		
I/J	0.570		
ddd	coplanarity 0.080		
bbb	mold flatness 0.100		

11 Revision History

Doc Rev.	Date	Comments
42384A	02/2016	Initial document release.

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:
AT32UC3A4256HHB-C1UR