Keywords: MAXQ,JTAG

APPLICATION NOTE 3966

Multiplexing JTAG Interface Pins on MAXQ Microcontrollers

Dec 18, 2006

Abstract: Often in embedded applications, every last port pin on a microcontroller is needed, leaving none to spare. Most MAXQ® microcontrollers with rewriteable internal program memory (such as flash memory or EEPROM) support a standardized JTAG/TAP interface (also known as the debug port) which is used by an external host to access in-circuit debugging or in-circuit programming (bootloader) functions. The pins which make up this interface are usually multiplexed with standard GPIO port-pin functions, which means that they are potentially available to the application instead of going to waste once the development phase has completed. This application note discusses methods of reusing these pins for general application purposes and presents considerations to keep in mind when doing so.

Overview

Often in embedded applications, every port pin on a microcontroller is needed for the application; none are left as spares. Developers do, however, have an option to address this problem. Most MAXQ microcontrollers with rewriteable internal program memory (such as flash memory or EEPROM) support a standardized JTAG/TAP interface (also known as the debug port), which is used by an external host to access in-circuit debugging or in-circuit programming (bootloader) functions. The pins used for this interface are usually multiplexed with standard GPIO port-pin functions, making them potentially available to the application once the development phase is complete. This application note explains how to reuse these interface pins for general application purposes. The note also identifies some situations to consider when multiplexnig those pins.

Stages of Application Development

During the development phase, the JTAG-compatible debug port provides a number of useful functions. First, the debug port allows the application to be loaded under the control of an external host (using a development environment such as MAX-IDE, Rowley CrossWorks, or the IAR Embedded Workbench®). This means that the application can be tested, modified, and quickly loaded again for the next testing cycle. Second, the debug port allows access to the in-circuit debugging features provided by the MAXQ architecture. These debugging features include the ability to read and write registers, step through program code a single instruction at a time, and view program, data and stack memory. Finally, the use of the bootloader and in-circuit debugger has little impact on the memory resources available to the application. This is because the in-circuit debugging functionality is implemented entirely in the MAXQ hardware and the Utility ROM.

Once the application is completed and tested, the in-circuit debugging functions are no longer needed. Moreover, in large-volume deployments where a reprogrammable MAXQ device is replaced by a

masked-ROM version, the in-circuit programming (bootloader) functions are also not needed, which means that the debug port is no longer serving any purpose and can be ignored...or used more creatively. When the number of GPIO port pins on a MAXQ device is limited and insufficient for an application, it may be especially helpful to reclaim the port pins dedicated to the JTAG-compatible debug port and make them available for general application use.

Reusing the Debug Port Pins

The following four pins are used to implement the JTAG-compatible debug port interface.
- TCK: Test Clock—Input to the MAXQ
- TMS: Test Mode Select—Input to the MAXQ
- TDO: Test Data Out—Output from the MAXQ
- TDI: Test Data In—Input to the MAXQ

These four pins are typically multiplexed with four GPIO port pins; the exact pins used for this purpose will vary from one MAXQ device to the next. By default, following a reset or power-on reset (POR) condition, the debug port is enabled, which means that the port pins are not available for general application use. To disable the debug port function and enable the port pins for general-purpose use, the TAP bit (SC.7) in the System Control register must be cleared to zero. The port pins are then controlled in the normal manner using the PD, PO, and PI registers.

Hardware Considerations

If the same hardware will be used during the application development and deployment phases, care must be taken that the hardware will operate properly when the port pins in the debug interface are used in either the GPIO mode or the JTAG/TAP mode. For example, when the pins are used in the JTAG/TAP mode, any external devices connected to these pins must release the pins to tri-state mode, thus allowing the host and the MAXQ to drive the signals on these lines. Additionally, devices connected to these lines must disregard any signals received on these lines which are driven by the host or MAXQ during in-circuit debug or bootloader operation. This is particularly true if responding to the signals could damage the device.

As an example, suppose one of the port pins was used both for the TCK signal (when used in JTAG/TAP mode) and to control a relay (in GPIO mode). When the device is being debugged using the JTAG interface, the TCK signal toggles rapidly, which in turn causes the relay to turn on and off and could damage external equipment connected to the relay. To prevent this, any external devices connected to pins from the JTAG/TAP interface should be disabled whenever the part will be operated in bootloader or in-circuit debugging mode. The external device might be disabled by using a jumper or another pin as an enable signal.

Software Considerations

Disabling the JTAG interface port is a simple matter. The TAP (SC.7) bit can be cleared to zero at any time, and doing so makes the port pins available to the application immediately. The natural inclination of the application developer may be to clear this bit at the beginning of the application code in order to set the proper operating mode for the application. Clearing the TAP bit this early, however, can cause issues during application development.

If the application will be running on a masked-ROM MAXQ device (which can never be reprogrammed), then there is no reason not to clear TAP to zero at the start of the application. In this case, the bootloader and in-circuit debugging functions will never be used, since the code is already programmed

in the device and cannot be altered.

However, for applications developed on a reprogrammable MAXQ device, the application software should always provide a few seconds delay before clearing the TAP bit and disabling the JTAG interface. If the TAP bit is cleared immediately following reset, this following sequence of events can result when trying to reload or debug the application:

1. The host drives active-low RESET low, placing the MAXQ in reset.
2. The host releases active-low RESET.
3. The MAXQ comes out of reset and begins running code, thus immediately shutting off the JTAG interface.
4. The host attempts to communicate with the part over the JTAG interface, but is unable to do so.

This sequence is similar to the problem caused by applications which immediately go into Stop mode or another extremely low-power mode following reset. The problem can be even worse on devices which allow the active-low RESET pin to be disabled as well. The actual reset behavior under these circumstances depends on the sequence of events triggered by the host, and whether the part is reset simply using the active-low RESET pin or by being powered on and off.

To avoid the above software issues, any application which shuts off the debug/TAP port or the active-low RESET pin should delay a few seconds on startup before shutting down the debug engine. This delay allows the external host to assume control of the MAXQ through the JTAG interface before this interface is turned off. Alternatively, the application could check the input level on another port pin (controlled with a jumper or pushbutton) to determine whether the JTAG port should be enabled or disabled.

Conclusion

The multiplexing function of the JTAG interface provided by the TAP (SC.7) bit on MAXQ microcontrollers allows the interface's four port pins to be used either for debugging/bootloader purposes or for general-purpose I/O. As long as certain hardware and software precautions are followed, these pins can be reused as additional resources for pin-limited systems and add flexibility when developing applications using MAXQ devices.

IAR Embedded Workbench is a registered trademark of IAR Systems AB.
MAXQ is a registered trademark of Maxim Integrated Products, Inc.

Related Parts		
MAXQ2000	Low-Power LCD Microcontroller	Free Samples

More Information
For Technical Support: http://www.maximintegrated.com/support
For Samples: http://www.maximintegrated.com/samples
Other Questions and Comments: http://www.maximintegrated.com/contact

Application Note 3966: http://www.maximintegrated.com/an3966
APPLICATION NOTE 3966, AN3966, AN 3966, APP3966, Appnote3966, Appnote 3966