

APPLICATION NOTE

ABSTRACT

This application note demonstrates how to use the LPC210x secondary JTAG interface while debugging the user application. The secondary JTAG interface provides the customer with 10 additional port pins, which would be otherwise, be allocated to the Embedded Trace Macrocell (ETM). The secondary JTAG interface can be used if the application only needs JTAG support for debugging.

AN10255

Philips LPC210x microcontroller family

Amitkumar Bhojraj

2004 Jan 07

Philips LPC210x microcontroller family

AN10255

INTRODUCTION

Before examining the secondary JTAG lets take a look at the LPC210x debug mode. The Debug Select (DBGSEL) and Returned Test Clock Output (RTCK) pins are used to enter the debug mode (primary JTAG and ETM). If DBGSEL is configured high on the rising edge of the CPU reset then pins P0.17–P0.31 are configured as debug pins. The RTCK pin must be high as the reset is released. The ARM7TDMI–S Debug Architecture uses the JTAG port along with the EmbeddedICE debug logic to provide on–chip debug support.

When JTAG and ETM are enabled, port pins P0.17–P0.31 are not useable by the application. The Primary JTAG interface uses pins P0.17–P0.21 and pins P0.22–P0.31 are used by the ETM (Embedded Trace Macrocell). The user may wish to debug the application using Primary JTAG only, but even then the bottom 10 port pins are not useable by the application.

The secondary JTAG interface is provided to free the ETM pins for use as port pins when debug with trace is not required. Under this interface user can debug the application and can have 10 additional port pins for the application, which would otherwise be used by the ETM. However, in this case the port pins used by the secondary JTAG interface will be pins P0.27–P0.31, which implies that all the remaining port pins from P0.0–P0.26 are useable by the application.

HOW TO CONFIGURE THE SECONDARY JTAG IN LPC210X

For configuring the secondary JTAG interface, the user needs to run a simple application from Flash on reset. If at least one of the DEBSEL or RTCK lines is low on reset then neither primary JTAG nor ETM pins are enabled. The code should map port pins (P0.27–P0.31) to alternate function 1, which is the secondary JTAG interface (Please refer to the Pin Configuration and Pin Connect Block chapters in the LPC2106/2105/2104 User Manual where the port pins P0.27–P0.31 are shown to be configurable to alternate function1). Since this application runs after reset the user can't switch to secondary JTAG in the same debug session.

Steps on how to switch to secondary JTAG are as follows:

1. Load the application in Flash using the software debugger and the primary JTAG interface. This application could also be loaded using an ISP utility (provided by some of our tool partners or by Philips itself)
2. Close the debugger or ISP utility (be sure to disconnect P.14 from ground)
3. Drive DEBSEL or RTCK low and connect port pins P0.27–P0.31 to the JTAG port (If your evaluation board supports the secondary JTAG interface then there should be a jumper that does the above)
4. Reset the part.
5. If the correct signature resides at 0x14 (More information in the Flash Memory System and Programming chapter in the LPC2106/2105/2104 User Manual) then user application in Flash will run and the port pins P0.27–P0.31 will be configured to secondary JTAG. The Philips Flash-programming tool and most of the debuggers handles the signature generation automatically.
6. User should then be able to debug the application using the secondary JTAG interface.

Software Example for configuring secondary JTAG

The application that runs from Flash at reset is provided in assembly and C. Since the interrupt vectors for ARM lie at 0x00–0x1C, this code must be linked to memory location 0x0. After the interrupt vectors, a few instructions are listed where the secondary JTAG interface is configured. The code has been developed in the ARM Development Suite (ADS) v1.2.

Assembly code:

```

; -----
;               Assembler Directives
; -----
        AREA IVT, CODE           ; New Code section
        CODE32                   ; ARM code

entry
; -----
        LDR        PC, =start
        LDR        PC, Undefined_Handler
        LDR        PC, SWI_Handler
        LDR        PC, Prefetch_Handler
        LDR        PC, Abort_Handler

; At 0x14 the user should insert a signature (checksum).
; This signature enables the bootloader to determine if
; there is valid user code in the Flash. Currently most of
; the Flash programming tools (debuggers and ISP utility)
; have this feature built-in so the end user need not worry
; about it. If the tool does not provide this feature then
; the value has to be computed manually and has to be
; inserted at 0x14. Details on computation of checksum
; could be found in the Flash programming chapter in the

```

Philips LPC210x microcontroller family

AN10255

```

; LPC2106/2105/2104 User Manual.

        DCD          ....
        LDR          PC, IRQ_Handler
        LDR          PC, FIQ_Handler
; -----
;           Exception Handlers
; -----
; The following dummy handlers do not do anything useful in
; this example. They are set up here for completeness.

Undefined_Handler          B Undefined_Handler

SWI_Handler                B SWI_Handler

Prefetch_Handler          B Prefetch_Handler

Abort_Handler             B Abort_Handler

IRQ_Handler                B IRQ_Handler

FIQ_Handler                B FIQ_Handler
; -----
;           Main code
; -----
start
        LDR SP=0x4.....      ; Set the Stack pointer for
                               ; the Supervisor mode
        LDR R0, JTAG2        ; Load R0 with 0x55400000
        LDR R1, PINSEL1     ; Load R1 with 0xE002C004
        STR R0, [R1]        ; Load PINSEL1 with 0x55400000
; -----
;           Allocate words in memory and assign values
; -----
JTAG2          DCD 0x55400000

PINSEL1        DCD 0xE002C004

                END

```

The user may have to modify the assembler directives depending upon the assembler being used. The function of the exception vectors here is to provide the signature for the bootloader at memory location 0x14, which enables the bootloader to detect that there is valid user code in the Flash memory.

On reset the first instruction to be executed would be

```
LDR PC, =start
```

which would branch to symbol start where SFR Pin Function Select Register 1 (Refer to Pin Connect block section in the LPC2106/2105/2104 User Manual) is loaded with 0x55400000. Port pins P0.27–P0.31 are now configured for alternate function 1 (secondary JTAG pins).

After loading this application into Flash and then performing the steps mentioned above, the debugger must be able to talk via the secondary JTAG interface.

C code (Interrupt Vector Table in assembly)

The code has been developed in the ARM Development Suite (ADS) v1.2. Only the relevant files are mentioned here, tool specific files are excluded. The code remains very much the same as above except that the main section where the secondary JTAG interface is configured is now written within C main (). The assembly code should reside from 0x0. The C file could be linked immediately after the interrupt vectors.

```

Interrupt Vector Table:
; -----
;           Assembler Directives
; -----
        AREA IVT, CODE      ; New Code section

```

Philips LPC210x microcontroller family

AN10255

```

CODE32          ; ARM code
IMPORT __main   ; symbol main not
                ; defined in this
                ; section

entry
; -----
LDR             PC, =start      ; jump to start
LDR             PC, Undefined_Handler
LDR             PC, SWI_Handler
LDR             PC, Prefetch_Handler
LDR             PC, Abort_Handler
; At 0x14 the user should insert a signature (checksum).
; This signature enables the bootloader to determine if
; there is valid user code in the Flash. Currently most of
; the Flash programming tools (debuggers and ISP utility)
; have this feature built-in so the end user need not worry
; about it. If the tool does not provide this feature then
; the value has to be computed manually and has to be
; inserted at 0x14. Details on computation of checksum
; could be found in the Flash programming chapter in the
; LPC2106/2105/2104 User Manual.
DCD             ....
LDR             PC, IRQ_Handler
LDR             PC, FIQ_Handler
; -----
;           Exception Handlers
; -----
; The following dummy handlers do not do anything useful in
; this example. They are set up here for completeness.
Undefined_Handler
    B           Undefined_Handler
SWI_Handler
    B           SWI_Handler
Prefetch_Handler
    B           Prefetch_Handler
Abort_Handler
    B           Abort_Handler
IRQ_Handler
    B           IRQ_Handler
FIQ_Handler
    B           FIQ_Handler
;-----
; Linked from the first instruction
;-----
start
    LDR SP,=0x4....      ; Setting up SP for SVC mode
    LDR LR,=__main       ; Jump to C main()
    MOV PC,LR

    END

C code:
#define PINSEL1 (*(volatile unsigned int *)0xE002C004)

int main()
{
    PINSEL1=0x55400000; // Configure Pins P0.27-P0.31
                       // to alternate function 1
                       // which sets up the
                       // secondary JTAG
}

```

Philips LPC210x microcontroller family

AN10255

CONCLUDING STATEMENTS

Combination of (DEBSEL+ RTCK) pins takes the microcontroller into debug mode and configures port pins P0.17–P0.31 as debug pins. If you wish to debug only through JTAG use the secondary JTAG interface (Port pins P0.27–P0.31) by running a simple application from Flash and by driving either or both DEBSEL and/or RTCK low. Hardware support is needed to use the secondary JTAG interface.

Philips LPC210x microcontroller family

AN10255

Definitions

Short-form specification — The data in a short-form specification is extracted from a full data sheet with the same type number and title. For detailed information see the relevant data sheet or data handbook.

Limiting values definition — Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 60134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.

Application information — Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Disclaimers

Life support — These products are not designed for use in life support appliances, devices or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes — Philips Semiconductors reserves the right to make changes, without notice, in the products, including circuits, standard cells, and/or software, described or contained herein in order to improve design and/or performance. Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no license or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Contact information

For additional information please visit
<http://www.semiconductors.philips.com>. Fax: +31 40 27 24825

For sales offices addresses send e-mail to:
sales.addresses@www.semiconductors.philips.com

© Koninklijke Philips Electronics N.V. 2004
All rights reserved. Printed in U.S.A.

Date of release: 01-04

Document order number:

9397 750 12482

Let's make things better.