

Programming Timed Port Events/Interrupts in Assembler

Version DAXS1100921R



Publication Date: 2010/09/21

Copyright © 2010 XMOS Ltd. All Rights Reserved.

1 Overview

This note explains a limitation of the XS1 port hardware that may cause unexpected behavior when programming in assembler. The limitation does not affect programs written in XC.

Problem

The problem is associated only with programs that use events or interrupts generated when an unbuffered port performs a timed unconditional input. The problem can only arise if a thread executes a program that:

- enables an unbuffered port to generate events or interrupts on timed unconditional inputs;
- enables another source such as a port, channel or timer to generate events or interrupts;
- handles a sequence of events or interrupts from the two sources without resetting the port time after handling each event or interrupt.

In this case, if the two sources of the events become ready within a short time interval, it is possible that the timed input event or interrupt is never generated. This is illustrated in *Section 2—Program in which a timed port may never generate an event*.

Solution

There are two simple ways to avoid this problem:

- Configure the port as a buffered port. In many cases this will be the best solution as there is little difference in behavior between an unbuffered port and a buffered port of the same width as the unbuffered port. This is illustrated in *Section 3—Program that uses a buffered port*.
- Use an instruction sequence that sets the input time each time the port handles an event or interrupt. Sequences of this kind are used by the XC Compiler. This is illustrated in *Section 4—Program that uses an alternative instruction sequence*.

Products affects

XS1-G, XS1-L, XS1-L2

Future products

This limitation will be removed in future versions of the XCore.

2 Program in which a timed port may never generate an event

```
// Test program that waits until two events have occurred:
// - one port's time to be equal to the time specified
// - another port's value to be equal to the value specified
#include <xs1.h>

// r0 contains port1
// r1 contains time for unconditional input
// r2 contains port2
// r3 contains data for condition
.align 4
getData:
    ldc r11, XS1_SETC_COND_NONE
    setc res[r0], r11 // set unconditional
    ldap r11, handler0
    setv res[r0], r11
    setpt res[r0], r1
    eeu res[r0] // enable events for port1

    ldc r11, XS1_SETC_COND_EQ
    setc res[r2], r11 // set condition equals
    ldap r11, handler1
    setv res[r2], r11
    setd res[r2], r3
    eeu res[r2] // enable events for port2

    ldc r10, 2 // number of events to wait for

waitloop:
    bf r10, done
    waiteu

handler0:
    sub r10, r10, 1
    edu res[r0]
    bu waitloop

handler1:
    sub r10, r10, 1
    edu res[r2]
    bu waitloop

done:
    retsp 0
```

3 Program that uses a buffered port

```
// Test program that waits until two events have occurred:
// - one port's time to be equal to the time specified
// - another port's value to be equal to the value specified
#include <xs1.h>

// r0 contains port1
// r1 contains time for unconditional input
// r2 contains port2
// r3 contains data for condition
.align 4
getData:
    ldc r11, XS1_SETC_BUF_BUFFERS
    setc res[r0], r11    // Example 2: set buffers

    ldc r11, XS1_SETC_COND_NONE
    setc res[r0], r11    // set unconditional
    ldap r11, handler0
    setv res[r0], r11
    setpt res[r0], r1
    eeu res[r0] // enable events for port1

    ldc r11, XS1_SETC_COND_EQ
    setc res[r2], r11    // set condition equals
    ldap r11, handler1
    setv res[r2], r11
    setd res[r2], r3
    eeu res[r2]         // enable events for port2

    ldc r10, 2          // number of events to wait for

waitloop:
    bf r10, done
    waitau

handler0:
    sub r10, r10, 1
    edu res[r0]
    bu waitloop

handler1:
    sub r10, r10, 1
    edu res[r2]
    bu waitloop

done:
    retsp 0
```

4 Program that uses an alternative instruction sequence

```
// Test program that waits until two events have occurred:
// - one port's time to be equal to the time specified
// - another port's value to be equal to the value specified
#include <xs1.h>

// r0 contains port1
// r1 contains time for unconditional input
// r2 contains port2
// r3 contains data for condition
.align 4
getData:
    ldc r11, XS1_SETC_COND_NONE
    setc res[r0], r11 // set unconditional
    ldap r11, handler0
    setv res[r0], r11
    eeu res[r0] // enable events for port1

    ldc r11, XS1_SETC_COND_EQ
    setc res[r2], r11 // set condition equals
    ldap r11, handler1
    setv res[r2], r11
    setd res[r2], r3
    eeu res[r2] // enable events for port2

    ldc r10, 2 // number of events to wait for

waitloop:
    setpt res[r0], r1 // Example 3: set port time before each event
    bf r10, done
    waiteu

handler0:
    sub r10, r10, 1
    edu res[r0]
    bu waitloop

handler1:
    sub r10, r10, 1
    edu res[r2]
    bu waitloop

done:
    retsp 0
```

5 Further Reading

For further information on the content covered by this advisory please see:

[The XMOS XS1 Architecture](#): Section 16—Events, Interrupts and Exceptions

Disclaimer

XMOS Ltd. is the owner or licensee of this design, code, or Information (collectively, the “Information”) and is providing it to you “AS IS” with no warranty of any kind, express or implied and shall have no liability in relation to its use. XMOS Ltd. makes no representation that the Information, or any particular implementation thereof, is or will be free from any claims of infringement and again, shall have no liability in relation to any such claims.

Copyright © 2010 XMOS Ltd. All Rights Reserved. XMOS and the XMOS logo are registered trademarks of XMOS Ltd in the United Kingdom and other countries, and may not be used without written permission. Company and product names mentioned in this document are the trademarks or registered trademarks of their respective owners. Where those designations appear in this document, and XMOS was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.