

# Address Span Extender Example

# Contents

- Example design
- Simulation Waves

Example design

# Address Span Extender Example

**Block Diagram**

address\_span\_extender\_0

clock, reset, windowed\_slave, cntl, avalon, expanded\_master, altera\_address\_span\_extender

**Data Path Properties**

- Datapath Width: 32 bits
- Byteenable Width: 4 bits
- Data Word Width: 4 bytes

**Address Properties**

- Expanded Master Byte Address Width: 38 bits
- Expanded Master Address Span: 256 gigabytes
- Slave Word Address Width: 28 bits
- Slave Address Span: 1 gigabyte
- Slave Address Shift: 2 bits

**Burst Properties**

- Burstcount Width: 4 bits
- Maximum burst: 8 words, 32 bytes

**Control Slave Properties**

- Control slave address width: 2 bits
- Number of sub-windows: 4
- Reset Default for Master Window: 0x0000000000000000 address
- Disable Slave Control Port: false
- Sub window span: 256 megabytes

**Bridge Slave Properties**

- Maximum Pending Reads: 1 read pending

**Address Span Extender Documentation**

**Address Span Extender**

**Name:** altera\_address\_span\_extender  
**Version:** 1.2  
**Author:** Altera Corp  
**Description:** Creates a windowed bridge to allow masters to access a larger address map.  
**Group:** Window Bridge

**Data Path Properties**

- Datapath Width:** Width of the slave and master data path.
- Byteenable Width:** Width of the slave and master byteenable port.
- Data Word Width:** Number of bytes in the data word.

**Address Properties**

- Expanded Master Byte Address Width:** Width of the master byte address port.
- Expanded Master Address Span:** Byte address span of master interface.
- Slave Word Address Width:** Width of the slave word address port.
- Slave Address Span:** Byte address span of slave interface.
- Slave Address Shift:** Number of address bits represented by the slave word width.

**Burst Properties**

- Burstcount Width:** Width of the slave and master burstcount port.
- Maximum burst:** Maximum words that a single burst can signal.
- Maximum burst:** Maximum bytes that a single burst can signal.

**Control Slave Properties**

- Control slave address width:** Width of the control slave address port.
- Number of sub-windows:** The slave port can represent 1 or more windows into the master address span.
- Reset Default for Master Window:** Reset default value for MASTER\_WINDOW
- Disable Slave Control Port:** MASTER\_WINDOW set by parameter
- Sub window span:** Byte address span of a single sub window.

**Bridge Slave Properties**

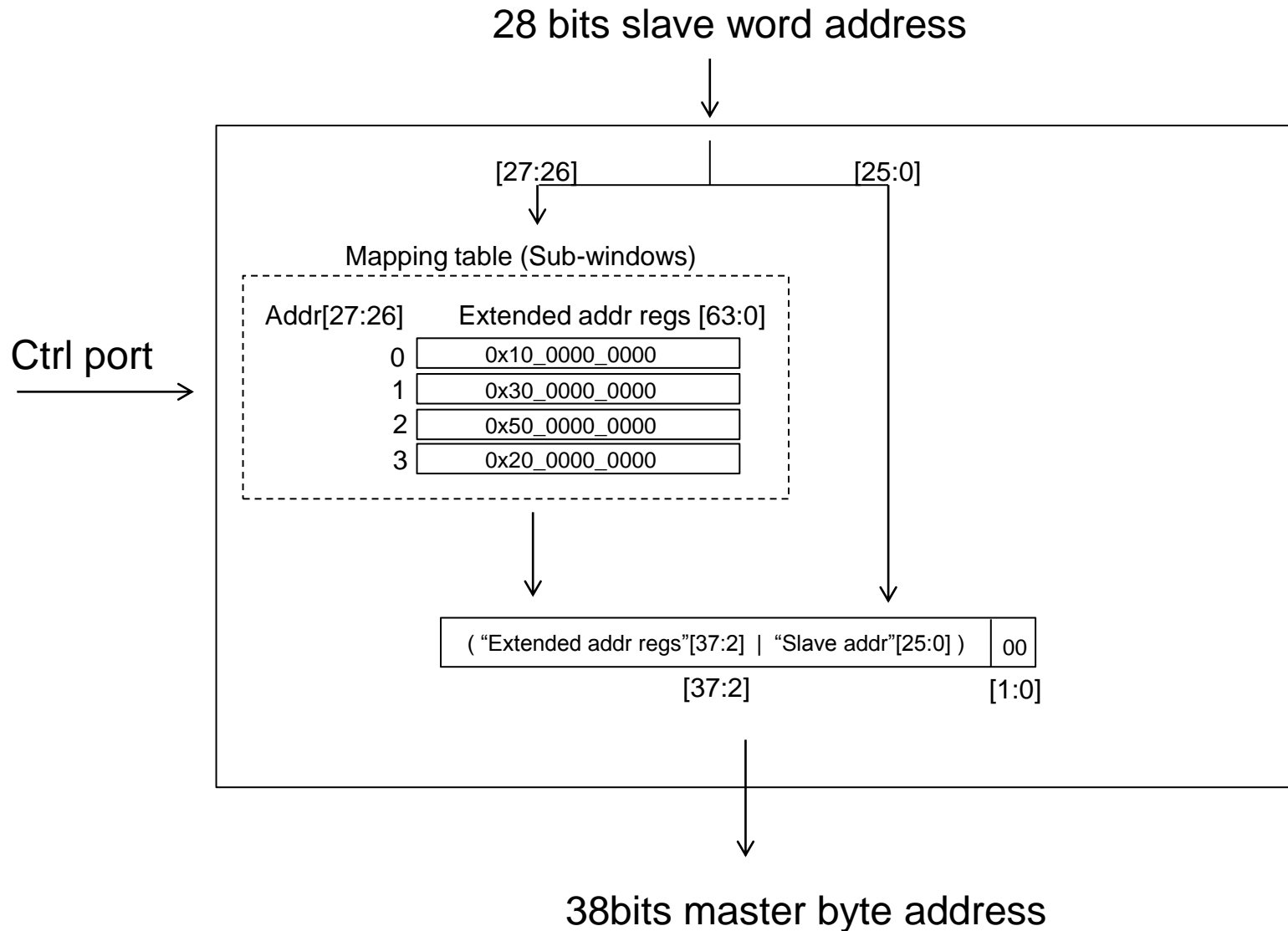
- Maximum Pending Reads:** Sets the bridge slave's maximumPendingReadTransactions property, in certain system configurations this value should be increased to improve performance.

Slave Word Address = 28 bits

Expanded Master Byte Address = 38 bits

Number of sub-windows = 4

# Address Span Extender Example (cont.)



# Address Span Extender Example (cont.)

System Contents	Address Map	Clock Settings	Project Settings	Instance Parameters	System Inspector	HDL Example	Generation	
Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		<ul style="list-style-type: none"> <li>clk_0</li> <li>clk_in</li> <li>clk_in_reset</li> <li>clk</li> <li>clk_reset</li> </ul>	<ul style="list-style-type: none"> <li>Clock Source</li> <li>Clock Input</li> <li>Reset Input</li> <li>Clock Output</li> <li>Reset Output</li> </ul>	<ul style="list-style-type: none"> <li>clk</li> <li>reset</li> </ul>	clk_0			
<input checked="" type="checkbox"/>		<ul style="list-style-type: none"> <li>nios2_qsys_0</li> <li>clk</li> <li>reset_n</li> <li>data_master</li> <li>instruction_master</li> <li>jtag_debug_module_reset</li> <li>jtag_debug_module</li> <li>custom_instruction_master</li> </ul>	<ul style="list-style-type: none"> <li>Nios II Processor</li> <li>Clock Input</li> <li>Reset Input</li> <li>Avalon Memory Mapped Master</li> <li>Avalon Memory Mapped Master</li> <li>Reset Output</li> <li>Avalon Memory Mapped Slave</li> <li>Custom Instruction Master</li> </ul>	<ul style="list-style-type: none"> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> </ul>	<ul style="list-style-type: none"> <li>clk_0</li> <li>[clk]</li> <li>[clk]</li> <li>[clk]</li> <li>[clk]</li> </ul>	IRQ 0	IRQ 31 ← x	
<input checked="" type="checkbox"/>		<ul style="list-style-type: none"> <li>onchip_memory2_0</li> <li>clk1</li> <li>s1</li> <li>reset1</li> </ul>	<ul style="list-style-type: none"> <li>On-Chip Memory (RAM or ROM)</li> <li>Clock Input</li> <li>Avalon Memory Mapped Slave</li> <li>Reset Input</li> </ul>	<ul style="list-style-type: none"> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> </ul>	<ul style="list-style-type: none"> <li>clk_0</li> <li>[clk1]</li> <li>[clk1]</li> </ul>	0x4000_8800	0x4000_8fff	
<input checked="" type="checkbox"/>		<ul style="list-style-type: none"> <li>address_span_extender_0</li> <li>clock</li> <li>reset</li> <li>windowed_slave</li> <li>expanded_master</li> <li>cntl</li> </ul>	<ul style="list-style-type: none"> <li>Address Span Extender</li> <li>Clock Input</li> <li>Reset Input</li> <li>Avalon Memory Mapped Slave</li> <li>Avalon Memory Mapped Master</li> <li>Avalon Memory Mapped Slave</li> </ul>	<ul style="list-style-type: none"> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> </ul>	<ul style="list-style-type: none"> <li>clk_0</li> <li>[clock]</li> <li>[clock]</li> <li>[clock]</li> <li>[clock]</li> </ul>	0x0000_0000	0x3fff_ffff	
<input checked="" type="checkbox"/>		<ul style="list-style-type: none"> <li>ctrl_master1_0</li> <li>m0</li> <li>clock</li> <li>reset</li> </ul>	<ul style="list-style-type: none"> <li>Avalon Memory Mapped Master</li> <li>Clock Input</li> <li>Reset Input</li> </ul>	<ul style="list-style-type: none"> <li>Double-click to export</li> <li>Double-click to export</li> <li>Double-click to export</li> </ul>	<ul style="list-style-type: none"> <li>[clock]</li> <li>clk_0</li> <li>[clock]</li> </ul>	0x0000	0x001f	

Data width:64 bits

Data width:32 bits

Master (byte) addr width=38 bits

Slave (word) addr width=28 bits  
(Byte address = 30 bits)

# Simulation Wave

# Setting sub-windows

Mapping table (Sub-windows)

addr	Extended addr regs [63:0]
0	0x10_0000_0000
1	0x30_0000_0000
2	0x50_0000_0000
3	0x20_0000_0000

/d_tb/d_inst/address_span_extender_0/avs_cntl_address	3	0	1	2	3
/d_tb/d_inst/address_span_extender_0/avs_cntl_write	0				
/d_tb/d_inst/address_span_extender_0/avs_cntl_writedata	0000002000...	0000...0000000100000000	0000000300000000	0000000500000000	0000000200000000



# Address Span Extender

In this case, slave data width = 32 bits (4bytes)  
Then slave word address width = (master bytes address >> 2)

Master byte address = 0x3c00\_0100



Extender Slave word address = 0xf00\_0040 (Sub-window# = [27:26] = 3)



Expanded Master byte address = 0x20\_0c00\_0100 (see mapping table on previous page)

