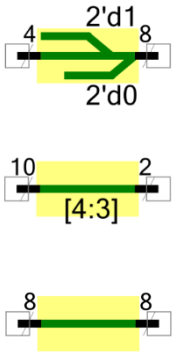
BusConnect
1.0

Features



- Interfaces two digital buses of mismatched widths.
- Has up to 32-bit width.
- Injects bits into a wider input.
- Extracts bits from a wider output.
- Facilitates routing as a dummy (empty) component.
- Virtual component – does not consume resources.
- Simplifies schematics.

General description

The BusConnect component is utility component, which provides easy interfacing of two digital buses of mismatching sizes. The component facilitates (i) connection of the narrow output to wide input (bits injection), (ii) connection of wide output to a narrow input (bits extraction). When input and output widths are equal, the BusConnect can serve as a dummy separator (empty pass-through component) for resolving certain bus routing errors. The BusConnect allows for parametrization of the bus routing inside custom-made components. The component is virtual, it doesn't consume system resources; it exists only at design time and is optimized out during the project build. Using BusConnect component saves space and simplifies schematic.

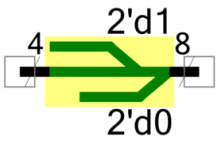
When to use BusConnect component

Component was developed for connecting 23-bit wide control bus of DDS24 custom component [1] to various digital sources, but it can be used with any other PSoC component which has digital input or output bus. Demo projects are provided.

Functional Description

When two digital buses of a mismatching sizes are being connected, the ambiguity arises in bits correspondence, with some lines being left unterminated[*], leading to compilation errors. As a result, tedious manual routing is required assigning input-to-output bits order and terminating unconnected lines. The BusConnect component allows easy assignment of the input-to output bits and hardcoded termination constants to the unconnected bits.

Bits injection



Component provides bits injection from the narrow bus into wider input. Consider an example when a digital output bus from 4-bit BasicCounter must be interfaced to the VDAC8 with 8-bit digital input bus (Figure 1). Such schematics shall not compile because of two kinds of errors: (i) the number of input and output bits do not match and (ii) some bits are having multiple drivers.

Figure 1. Example of the schematic that shall not build.

Since there are more input bits than sources available, it has to be explicitly stipulated which input bits are being connected to data source and which are being terminated. For example, let's decide that bits [5:2] are connected to the source BasicCounter, the least significant bits (LSBs) [1:0] are hardcoded with digital "1" each, and the remaining most significant bits (MSBs) [7:6] are hardcoded with digital "0" (Figure 2). Such arrangement shall produce 16-step voltage ramp with fixed offset of 3.

[*] The unconnected bits of the input bus can't be left free-hanging.

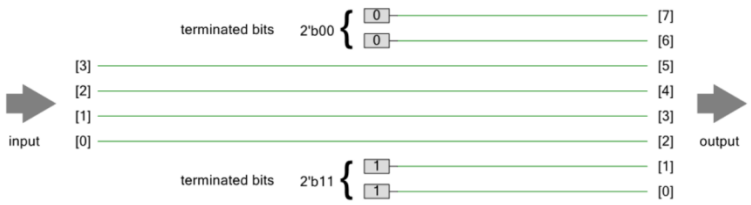


Figure 2. Example of bits [5:2] being connected to the data source, the LSB's value hardcoded to 2'b11 (0x03) and MSB's are hardcoded to 2'b00 (0x00).

In Cypress Creator IDE such assignment can be accomplished by using regular bus indexing tools and digital constants (Figure 3).

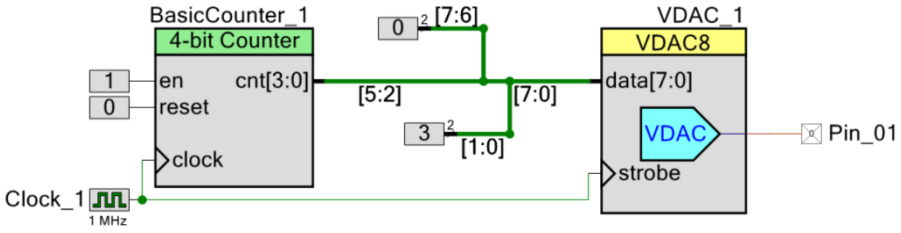


Figure 3. Bits injection using bus standard indexing tools and digital constants.

In this example, the LSBs bus [1:0] of width 2, is being terminated with digital constant value 0x03 (binary 2'b11); input bits [2:5] are being connected to the BasicCounter, and MSBs bits [7:6] are terminated with digital constant value of 0x00 (binary 2'b00).

Using the BusConnect the above schematic can be simplified, while preserving all visual information (Figure 4).

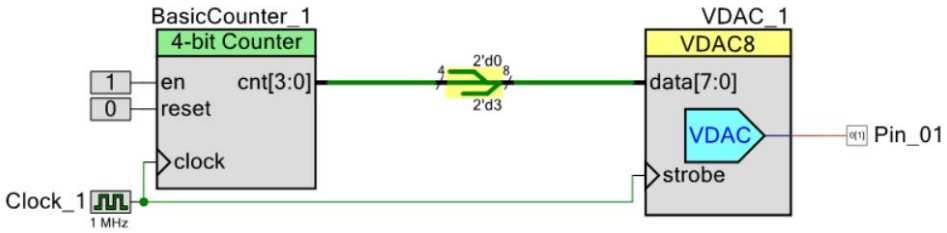


Figure 4. Bits injection using BusConnect component.

All tree representation of bus connection on Figures 2, 3 and 4 are equivalent and summarized on Figure 5. The BusConnect symbol displays the widths of all buses, and termination constants for the MSBs and LSBs using standard decimal notation {width}'d{value}.

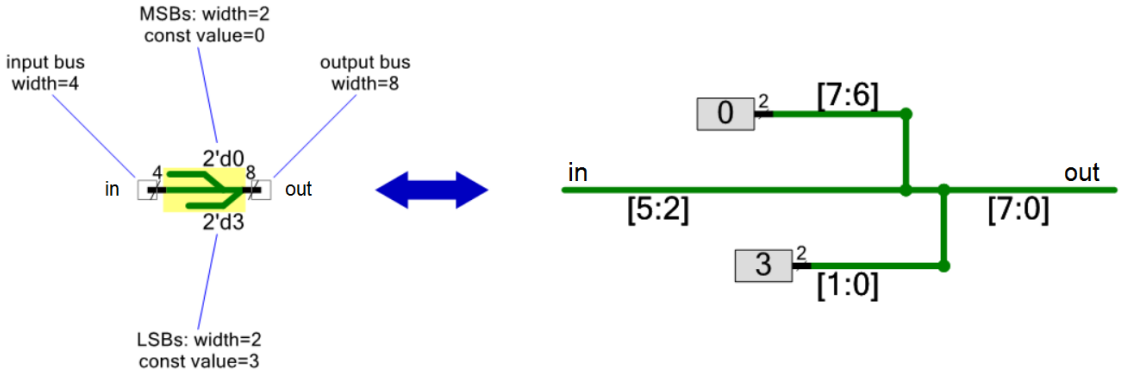
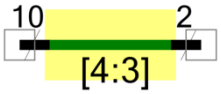


Figure 5. Equivalent representation of the BusConnect interface.

Bits extraction



Component provides bits extraction from a wide source into the narrow input. In the example below the BusConnect interfaces top 8 bits of the 10-bit BasicCounter to the VDAC8 (Figure 6). The same can be accomplished using Creator standard bus indexing tools; however, using the BusConnect has advantage when schematic is encapsulated inside a custom component. In such case the bus parameters can be easily passed from the top component, which is not possible with standard indexing tools.

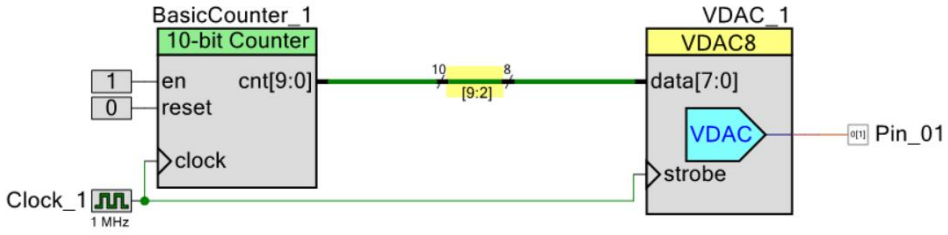
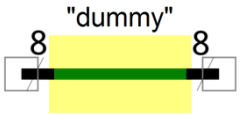


Figure 6. Partial bits extraction using BusConnect component.

Dummy pass-through



By selecting same input and output widths the BusConnect becomes a "dummy" (empty pass-through) component. In this case the component seemingly does not do anything: it freely passes across all signals and is automatically optimized out during the project build. However, in such form the component can facilitate CyFitter automatic routing and to avoid errors in certain cases. See **Appendix 1** for details. Another dedicated component, the Dummy, is also available for this specific purpose [2].

Parametrization

One useful feature of the component is bus routing parametrization, when actual schematic design occurs during build process according to the parameters passed from the top level schematics. This is useful for custom component creation, when bus customization inside the component is not possible the other ways. For example, in the stepped ramp generator component (Figure 7) the number of steps can be controlled by the Width parameter of the BasicCounter (*steps = 2*^{Width}). To build successfully, the BusConnect parameters *inp_width* and *LSB_width* should be set to *Width* and *8-Width* correspondingly.

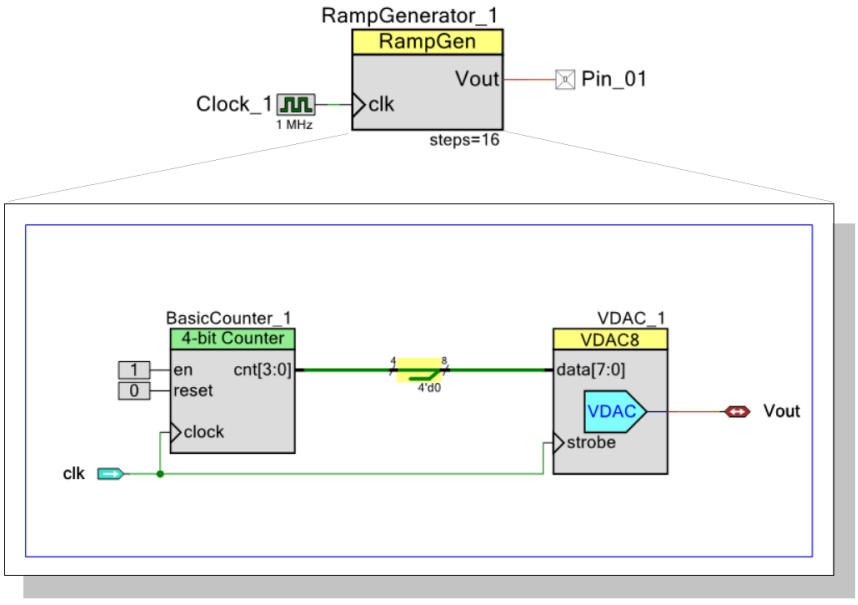


Figure 7. Using BusConnect for custom component schematic parametrization.

Input-output connections

inp – input bus

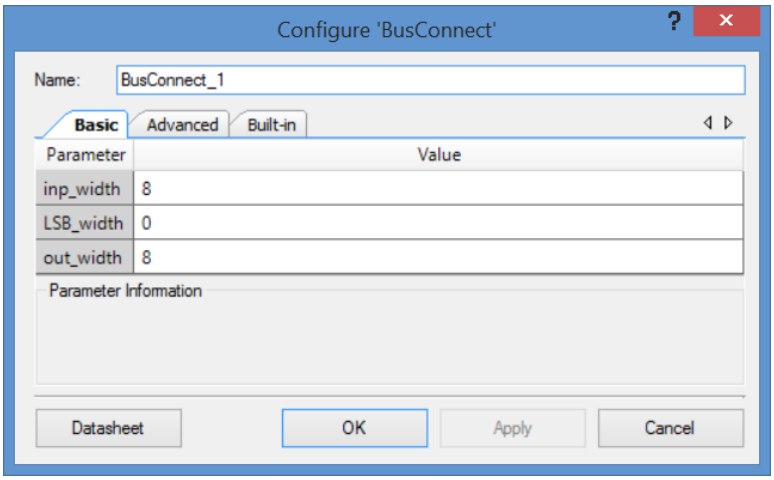
This pin is digital input bus. The bus width is user-selectable, valid range 1 to 32. The pin is always visible, and must be connected to a valid digital source.

outp – output bus

This pin is digital output bus. The bus width is user-selectable, valid range is 1 to 32. This pin is always visible. The pin doesn't have to be connected.

Parameters and Settings

Basic dialog provides following parameters:



inp_width (uint8)

Input bus width. Valid range is from 1 to 32. Default value is 8. When *inp_width* is less then *outp_width*, the component is configured for bits injection. When *inp_width* is larger than *outp_width*, the component is configured for bits extraction. When *inp_width* equals *outp_width*, the component becomes a dummy pass-through.

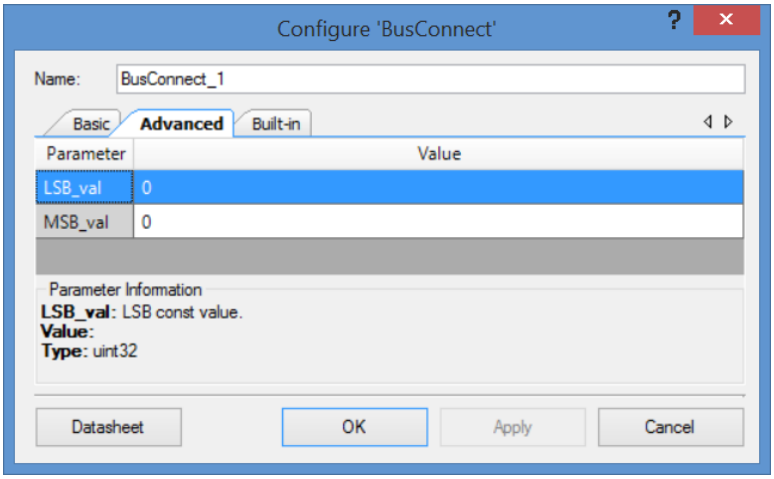
LSB_width (uint8)

Width of the LSBs group of unconnected pins. Valid range is from 0 to (*out_width – inp_width*). Default value is 0 (no LSBs bus). The MSBs width is automatically calculated from the values of the input, output and LSBs width as: (*out_width – inp_width – LSB_width*). The symbol appearance automatically updated to display status of the LSBs and MSBs buses.

out_width (uint8)

Output bus width. Valid range is from 1 to 32. Default value is 8.

Advanced dialog provides following parameters:



LSB_val (uint32)

Hardcoded constant assigned to the LSBs group of unconnected pins. Default value is 0. The value assigned should not exceed the maximum allowed by the number of the available bits.

MSB_val (uint32)

Hardcoded constant assigned to the MSBs group of unconnected pins. Default value is 0. The value assigned should not exceed the maximum allowed by the number of the available bits.

Application Programming Interface

The component does not have associated API. All configuration parameters are set at design time and can't be altered during the run-time.

Resources

The component doesn't consume system resources. It is virtual; it exists only at design time and is automatically optimized out during the project build.

Performance

The component doesn't affect system performance. The component is not device-specific and compatible with PSoC4 and PSoC5.

Sample Firmware Source Code

Several component application ideas are provided in the **Appendix 1**. Demo projects provided.

Component Changes

Version	Description of changes	Reason for changes/impact
0.0	Version 0.0 is the first beta release of the BusConnect component	
1.0	Added bits extraction option. Datasheet provided.	

References

1. DDS24: 24-bit DDS arbitrary frequency generator, https://community.cypress.com/thread/13628
2. The Dummy: empty component for digital bus routing, https://community.cypress.com/thread/13152
3. Chirp burst generator for ultrasound range finder, https://community.cypress.com/thread/51313
4. ADC_SAR with hardware output bus, https://community.cypress.com/message/185064#185064

Appendix 1

ADC_SAR – Filter – VDAC

Example of using BusConnect component for interfacing ADC_SAR to the FIFOin custom component[(*)] is shown on Figure 8. In this example the BusConnect is used for scaling up the ADC digital output[(†)]. Here the LSB bits [0:2] and MSB bit [15] are padded with zeros. The ADC output bits [0:11] are being shifted up by 3 bits, multiplying ADC digital output by 8. Digitally amplified signal is then transferred to the Filter by DMA. The FIFOin here serves as 16-bit register with DMA access.

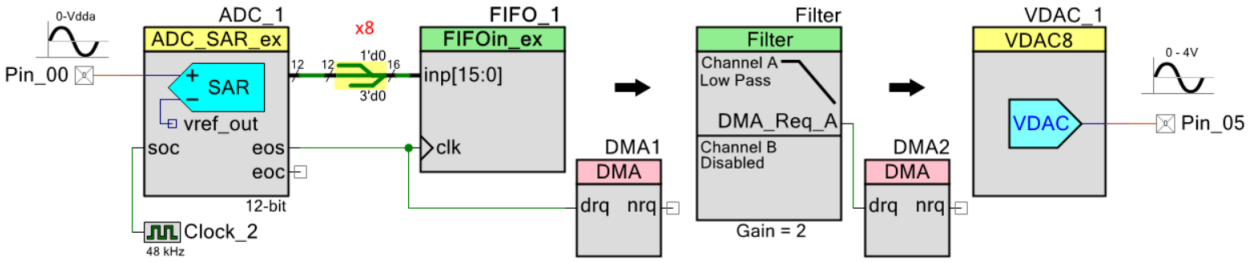


Figure 8. ADC_SAR – Filter – VDAC example using BusConnect.

This arrangement helps to resolve a mismatch between 12-bit ADC output and the input stage register of the Filter, expecting 16-bit data. If the ADC data were transferred directly to the Filter, the filter output will be only 1/16 of the maximum scale. This presents no issue when Filter output is being consumed by the processor, but when Filter output is transferred to VDAC8 by DMA the output signal appears to be too weak. Using BusConnect resolves the issue.

Voltage Controlled Oscillator

Example of using BusConnect for interfacing ADC_SAR and DDS24 custom component [1] is shown on Figure 9. Schematic depicts a Voltage Controlled Oscillator (VCO), where DDS24 tuning word is hardware-controlled by the ADC digital output. Here BusConnect is used for changing the VCO gain and offset. The LSBs [0:4] and MSBs [17:23] bits are padded with zeros (no offset), and the ADC output bits [0:11] are shifted up by 5 bits, magnifying ADC digital output by 32.

[*] FIFOin_ex is a modified version of the Brad Budlong custom component FIFOIn (original link is no longer valid).
[†] ADC_SAR_ex is customized version of the stock ADC_SAR component with hardware output bus [4].

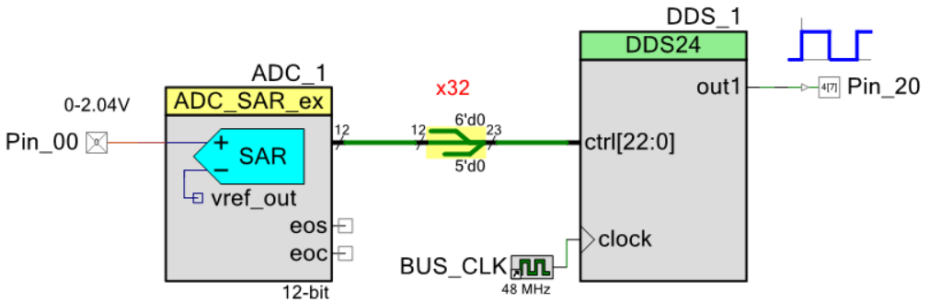


Figure 9. Adjusting VCO gain using BusConnect.

The DDS24 output frequency is directly proportional to the ADC input voltage. By selecting the ADC input scale, DDS base clock frequency and by shifting DDS control bits the VCO gain coefficient can be adjusted from approx. 5 kHz/V to 1.5 MHz/V. As shown:

$$K_{VCO} = \frac{4095}{2.048\ V} \times 2^5 \times \frac{48\ MHz}{2^{24}} = 183\ kHz/V$$

Chirp generator

All-hardware chirp (frequency sweep) generator for ultrasound range detection using BusConnect is shown on Figure 10. It produces a burst of the frequency sweep between 38 kHz to 43 kHz in 5 ms for PZT excitation, followed by the 5 ms of quiet period for detection of the reflected signal [3]. The number of ramp steps (N=2^6) is controlled by the *inp_width*. The ramp starting and ending frequencies are controlled by the MSB constant (8 = 2'b00000111), *LSB_width* (9), and DDS clock (2.8 MHz):

$$F_{start} = 2'\text{b}00000111\_000000\_000000000 \times 2.8\ MHz/2^{24} = 38.28\ kHz,$$
$$F_{end} = 2'\text{b}00000111\_111111\_000000000 \times 2.8\ MHz/2^{24} = 43.66\ kHz.$$

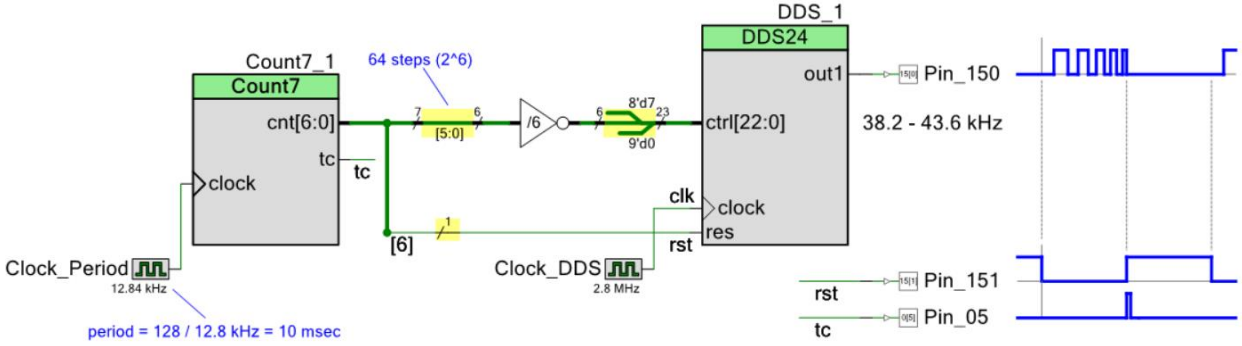


Figure 10. Hardware chirp generator using BusConnect.

BusConnect as dummy component

By selecting same input and output widths the BusConnect becomes a "dummy" (empty pass-through) component. The component freely passes all signals across and is automatically eliminated during project build. In such form the component can be used to facilitate CyFitter automatic routing, avoiding build errors in certain situations.

Consider schematic depicted on Figure 11. In this example all bits within the bus are freely passed from one component to another, except for the LSB, which is being manipulated through external digital input. Though all range assignments seems to be in place, attempt to build the project terminates with error: 'Multiple drivers on signal, "Net_…", bit(s) "0" '.

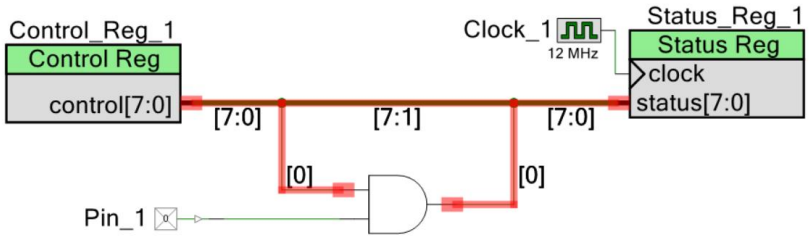


Figure 11. Example of schematic that shall not build.

By adding extra "dummy" component to the bus, the issue is resolved and project builds without errors (Figure 12). Note that the BusConnect does not affect signals in any way and is eliminated by the fitter during optimization.

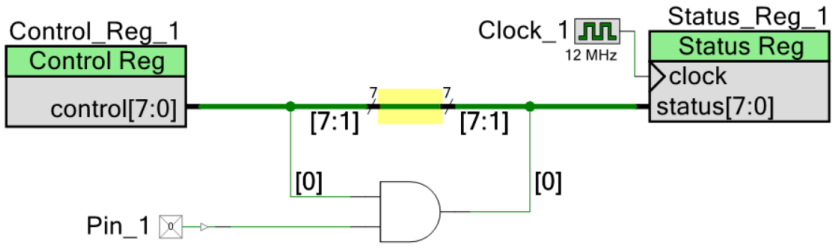


Figure 12. Routing facilitated using BusConnect component.