

(기계로 한국어 번역)

Getting started with the FPGA demo bundle for Altera

Xillybus Ltd.

www.xillybus.com

Version 3.2

이 문서는 영어에서 컴퓨터에 의해 자동으로 번역되었으므로 언어가 불분명할 수 있습니다. 이 문서는 원본에 비해 약간 오래되었을 수 있습니다.

가능하면 영문 문서를 참고하시기 바랍니다.

This document has been automatically translated from English by a computer, which may result in unclear language. This document may also be slightly outdated in relation to the original.

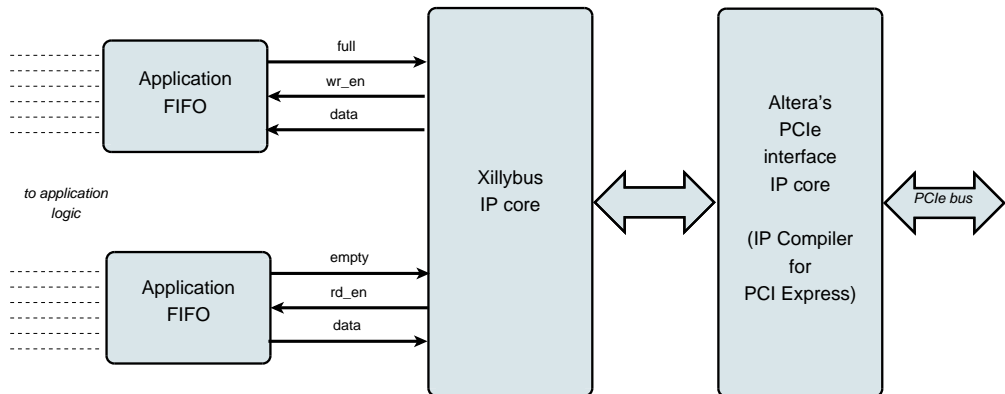
If possible, please refer to the document in English.

1 소개	3
2 전제 조건	5
2.1 하드웨어	5
2.2 FPGA 프로젝트	6
2.3 개발 소프트웨어	6
2.4 FPGA design 사용 경험	7
3 demo bundle의 implementation	8
3.1 개요	8
3.2 파일 개요	8
3.3 demo bundle 구축	9
3.3.1 프로젝트 열기	9
3.3.2 PCIe 블록 만들기(Arria II 및 series-IV FPGAs)	9
3.3.3 design 파일의 Compilation	14
3.4 FPGA 프로그래밍	15
4 수정	16
4.1 맞춤형 logic와 통합	16
4.2 Altera의 PCIe IP / Megafunction에서 변경하기	17
4.3 사용자 지정 프로젝트에 포함	17
4.4 다른 보드 사용	19
4.4.1 일반적인	19
4.4.2 장치 제품군	19
4.4.3 Pin placements	19
4.4.4 PCIe만 해당: Clocking	19
4.4.5 XillyUSB로 작업하기	20
5 문제 해결	22
5.1 implementation 중 오류	22
5.2 PCIe 하드웨어 문제	22

1

소개

Xillybus는 FPGA와 Linux 또는 Microsoft Windows를 실행하는 host 간의 데이터 전송을 위한 DMA 기반 종단 간 솔루션입니다. FPGA logic의 설계자와 소프트웨어 프로그래머에게 간단하고 직관적인 인터페이스를 제공합니다.



위에 표시된 것처럼 FPGA의 application logic은 표준 FIFOs와만 상호 작용하면 됩니다. 예를 들어, 다이어그램에서 하위 FIFO에 데이터를 쓰는 것은 Xillybus IP core가 FIFO의 다른 쪽 끝에서 데이터를 전송할 수 있음을 감지하게 합니다. 곧 IP core는 FIFO에서 데이터를 읽고 host로 보내 userspace software에서 읽을 수 있도록 합니다. 데이터 전송 메커니즘은 FIFO와만 상호 작용하는 FPGA의 application logic에 투명합니다.

반면 Xillybus IP core는 PCI Express의 Transport Layer level을 활용하여 데이터 흐름을 구현하여 TLP 패킷을 생성 및 수신합니다. 하위 계층의 경우 개발 도구의 일부인 Altera의 공식 PCIe core에 의존하며 추가 라이선스가 필요하지 않습니다(Quartus의 Web/ Lite Edition을 사용하는 경우에도).

컴퓨터의 응용 프로그램은 named pipes처럼 동작하는 device files와 상호 작용합니다. Xillybus IP core 및 driver는 FPGA의 FIFOs와 host의 관련 device files 간에 데이터를

효율적이고 직관적으로 전송합니다.

XillyUSB에서 MGT transceiver는 위에서 언급한 PCIe 인터페이스 대신 데이터 전송에 사용되는 USB 3.0 인터페이스를 구현하는 데 사용됩니다.

IP core는 온라인 웹 애플리케이션을 사용하여 고객의 사양에 따라 즉시 구축됩니다. streams의 수, 방향 및 기타 속성은 design의 대역폭 성능, 동기화 및 단순성 간의 최적 균형을 달성하기 위해 고객이 정의합니다. 이 가이드에 설명된 대로 demo bundle의 준비 단계를 거친 후 <https://xillybus.com/custom-ip-factory>에서 사용자 지정 IP core를 빌드하고 다운로드하는 것이 좋습니다.

이 가이드는 실제 애플리케이션 시나리오 테스트를 위해 사용자 제공 데이터 소스 및 데이터 소비자에 연결할 수 있는 Xillybus IP core와 함께 FPGA를 빠르게 설정하는 방법을 설명합니다. IP core는 웹사이트에서 다운로드할 수 있는 demo bundle에 포함되어 있습니다.

이름에도 불구하고 demo bundle은 데모 키트가 아니라 유용한 작업을 있는 그대로 수행할 수 있는 완전한 기능의 starter design입니다.

궁금한 분들을 위해 Xillybus 구현 방법에 대한 간략한 설명은 [Xillybus host application programming guide for Linux](#) 또는 [Xillybus host application programming guide for Windows](#)의 부록 A에서 찾을 수 있습니다.

2

전제 조건

2.1 하드웨어

Xillybus는 PCI Express용 Altera의 hardware IP block에 의존하므로 이 구성 요소가 있는 모든 Altera 장치에서 사용할 수 있습니다. 지원되는 FPGA 제품군:

- Arria II GX/GZ
- Cyclone IV GX
- HardCopy IV GX
- Stratix IV GX
- Arria V GX/GT/SX/ST
- Cyclone V GX/GT/SX/ST
- Stratix V GS/GX/GT
- Arria 10 GX/GT/SX
- Cyclone 10 GX
- H-tile 또는 L-tile이 있는 Stratix 10

XillyUSB는 Cyclone 10 GX에서만 지원됩니다(LP 제품군은 지원되지 않음).

Xillybus FPGA demo bundle은 다운로드 페이지에 나열된 대로 여러 보드 및 장치와 함께 작동하도록 패키징되어 있습니다(아래 2.2 섹션 참조).

다른 보드의 소유자는 pin placements에서 필요한 변경을 수행하고 MGT의 reference clock이 제대로 처리되는지 확인한 후 자신의 하드웨어에서 demo bundle을 실행할 수 있

습니다. 경험이 많은 FPGA 엔지니어라면 누구나 쉽게 이해할 수 있을 것입니다. 이에 대한 자세한 내용은 4.4 섹션을 참조하십시오.

2.2 FPGA 프로젝트

Xillybus demo bundle은 Xillybus 사이트의 다운로드 페이지에서 다운로드할 수 있습니다. PCIe 기반 cores의 경우:

<https://xillybus.com/pcie-download>

그리고 XillyUSB의 경우:

<https://xillybus.com/usb-download>

demo bundle에는 간단한 테스트를 위한 Xillybus IP core의 특정 구성이 포함되어 있습니다. 따라서 특정 응용 프로그램에 대해서는 상대적으로 성능이 좋지 않습니다.

사용자 지정 IP cores는 IP Core Factory 웹 애플리케이션을 사용하여 구성, 자동 구축 및 다운로드할 수 있습니다. 이 도구를 사용하려면 <https://xillybus.com/custom-ip-factory>를 방문하십시오.

Xillybus IP core를 포함하여 다운로드한 번들은 “evaluation”라는 용어와 합리적으로 일치하는 한 무료로 사용할 수 있습니다. 여기에는 최종 사용자 designs에 core 통합, 실제 데이터 실행 및 현장 테스트가 포함됩니다. core 사용의 유일한 목적이 특정 애플리케이션에 대한 기능과 적합성을 평가하는 것이라면 core 사용 방법에는 제한이 없습니다.

2.3 개발 소프트웨어

FPGA 제품군에 따라 Xillybus의 demo bundle(및 Xillybus와 관련된 다른 designs)의 implementation에 권장되는 도구가 아래에 나열되어 있습니다.

PCIe용 Xillybus :

- series-IV 및 Arria II의 FPGAs: Quartus 12.0 이상.
- Arria 10 및 Cyclone 10의 경우: Quartus Prime 17.1 이상. Standard 및 Pro 에디션이 모두 지원됩니다.
- Stratix 10의 경우: Quartus Pro 19.2 이상.
- 기타 모든 FPGA 제품군: Quartus II, 버전 15.0 이상.

XillyUSB :

- Cyclone 10의 경우: Quartus Pro 17.1 이상을 사용해야 합니다.

이 소프트웨어는 Altera의 웹사이트(<https://www.altera.com>)에서 직접 다운로드할 수 있습니다.

무료로 제공되는 Quartus의 Web/Lite Edition은 여러 FPGA 제품군, 특히 Cyclone 장치를 지원합니다.

Xillybus의 implementation은 Quartus에서 제공하는 일부 IP cores에 의존합니다. Quartus의 모든 에디션은 추가 라이선스 없이 이러한 IP cores를 지원합니다.

2.4 FPGA design 사용 경험

design이 demo bundles 목록에 나타나는 보드용인 경우 FPGA design에 대한 이전 경험이 없어도 demo bundle이 FPGA에서 작동합니다. 다른 보드를 사용하는 경우 Altera의 도구 사용에 대한 지식, 특히 pin placements 및 clocks에 대한 정의가 필요합니다.

demo bundle을 최대한 활용하려면 logic design 기술을 잘 이해하고 HDL 언어(Verilog 또는 VHDL)를 마스터해야 합니다. 그럼에도 불구하고 Xillybus demo bundle은 실험할 간단한 starter design을 제공하므로 이러한 학습을 위한 좋은 출발점입니다.

3

demo bundle의 implementation

3.1 개요

Xillybus의 demo bundle의 implementation와 bit stream 파일(SOF)을 얻는 세 가지 가능한 방법이 있습니다.

- 번들에 있는 프로젝트 파일을 그대로 사용합니다. 이것은 가장 간단한 방법이며 demo bundles 목록에 나타나는 보드로 작업할 때 적합합니다.
- 다른 FPGA와 일치하도록 파일 수정. 이것은 다른 보드 및/또는 다른 FPGAs와 함께 작업할 때 적합합니다. 이에 대한 자세한 내용은 단락 4.4를 참조하십시오.
- Quartus 프로젝트를 처음부터 설정합니다. demo bundle을 기존 application logic와 통합할 때 필요할 수 있습니다. 4.3 단락에 자세한 내용이 있습니다.

이 섹션의 나머지 부분에서는 첫 번째 작업 절차가 자세히 설명되어 있으며 가장 간단하고 가장 일반적으로 선택됩니다. 다른 두 가지 작업 절차는 첫 번째 절차를 기반으로 하며 차이점은 위에 제공된 단락에 자세히 설명되어 있습니다.

중요한:

평가 번들은 성능보다는 단순성을 위해 구성됩니다. 지속적이고 지속적인 데이터 흐름이 필요한 애플리케이션, 특히 고대역폭의 경우 훨씬 더 나은 결과를 얻을 수 있습니다. 이러한 시나리오의 경우 사용자 지정 IP core는 웹 응용 프로그램과 함께 쉽게 구축 및 다운로드됩니다.

3.2 파일 개요

번들은 5개의 디렉토리로 구성됩니다.

- core- Xillybus IP core는 여기에 저장됩니다.
- instantiation templates- core용 instantiation templates 포함(Verilog 및 VHDL)
- verilog- demo bundle에 대한 프로젝트 파일과 Verilog의 소스를 포함합니다('src' 하위 디렉토리에 있음).
- vhd- demo bundle에 대한 프로젝트 파일과 VHDL의 소스를 포함합니다('src' 하위 디렉토리에 있음).
- pcie_core(PCle 번들만 해당) - Altera의 PCle IP core는 번들의 나머지 부분을 빌드하기 전에 여기에 빌드됩니다.
- quartus-essentials(XillyUSB 번들만 해당) - Verilog 및 VHDL 프로젝트에 공통적인 다양한 파일.

각 demo bundle은 demo bundle을 다운로드한 사이트의 웹 페이지에 나열된 대로 특정 보드용입니다. 다른 보드를 사용하거나 특정 구성 저항이 보드에서 추가 또는 제거된 경우 constraints 파일을 그에 따라 편집해야 합니다.

또한 vhd 디렉토리에는 Verilog 파일이 포함되어 있지만 크게 변경할 필요는 없습니다.

Xillybus의 IP core와 application logic 사이의 인터페이스는 xillydemo.v 파일 또는 xillydemo.vhd 파일(해당 'src' 하위 디렉토리)에서 이루어집니다. 자신의 데이터로 Xillybus를 사용하기 위해 편집하는 파일입니다.

3.3 demo bundle 구축

3.3.1 프로젝트 열기

기본 설정에 따라 'verilog' 또는 'vhd' 하위 디렉토리에서 'xillydemo.qpf' 파일을 두 번 클릭합니다. Quartus는 올바른 설정으로 프로젝트를 시작하고 엽니다.

series-V(예: Cyclone V) 또는 series-10(예: Arria 10) FPGA가 사용되는 경우 단락 3.3.3로 계속 진행합니다.

그렇지 않으면 다음에 설명된 대로 PCle 블록을 빌드하십시오.

3.3.2 PCle 블록 만들기(Arria II 및 series-IV FPGAs)

중요한:

이 단락은 series-V FPGAs 이상과 관련이 없습니다.

12.0 이후의 Quartus 버전을 사용하는 경우 megafunction variation 파일을 수동으로 편집해야 합니다. 그렇지 않으면 MegaWizard Plug-In Manager는 이 파일을 수락하지 않습니다.

편집이 필요한 경우 텍스트 편집기로 pcie_core/ 디렉토리(예: pcie_core/ pcie_s4_4x.v)에 있는 파일을 엽니다. 이전 Quartus 버전의 모든 항목을 사용된 것으로 교체하십시오. 버전 번호를 변경하기만 하면 됩니다(예: 12.0을 15.0으로 교체).

일반적으로

```
// megafunction wizard: %IP Compiler for PCI Express v12.0%
```

및

```
// Retrieval info: <MEGACORE title="IP Compiler for PCI Express" version="12.0"
```

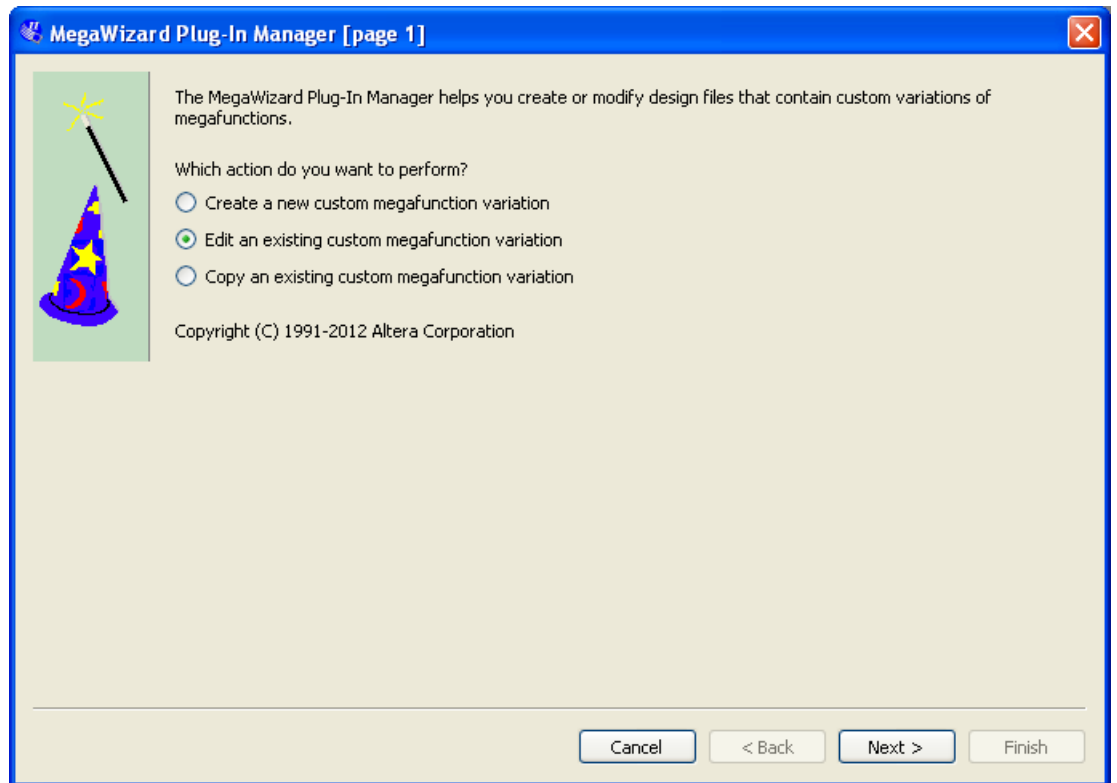
및 가능하면 다음과 같거나 유사한 라인을 수정해야 합니다.

```
// Generated using ACDS version 12.0 178
```

Quartus 내에서 MegaWizard Plug-In Manager 실행:

- Quartus 14 이상: Command Prompt 창(또는 Linux에서 terminal)을 엽니다. Quartus의 유틸리티가 execution path(일반적으로 /some/path/quartus/bin/)에 있는지 확인하십시오. "qmegawiz"를 입력하여 MegaWizard Plug-In Manager를 시작합니다.
- Quartus 13 이하: 도구 메뉴에서 MegaWizard Plug-In Manager를 선택합니다.

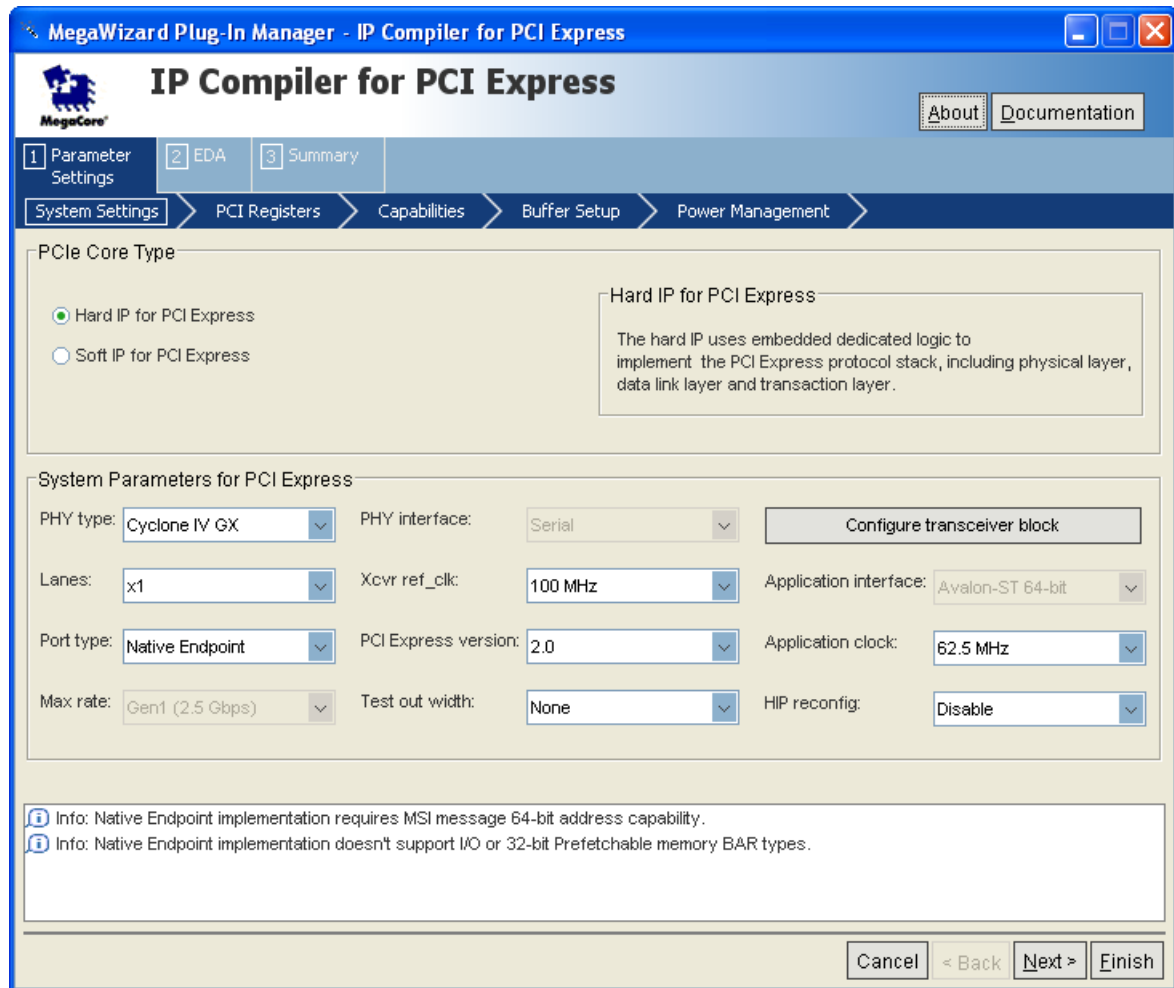
다음(또는 유사한) 창이 나타납니다.



“Edit an existing custom megafunction variation”을 선택하고 다음을 클릭합니다.

다음으로, 창(여기에 표시되지 않음)은 편집할 사용자 지정 megafunction variation 파일을 요청합니다. `pcie_core` 디렉터리에서 `pcie_c4_1x.v`(또는 이와 유사한) 파일을 선택합니다(적합한 파일 하나만 표시됨). 일반적으로 시작 지점에서 위로 한 디렉터리를 탐색하면 들어갈 디렉터리가 나타납니다.

“Loading MegaWizard...”라는 알림 후 다음과 같은 창이 나타납니다.

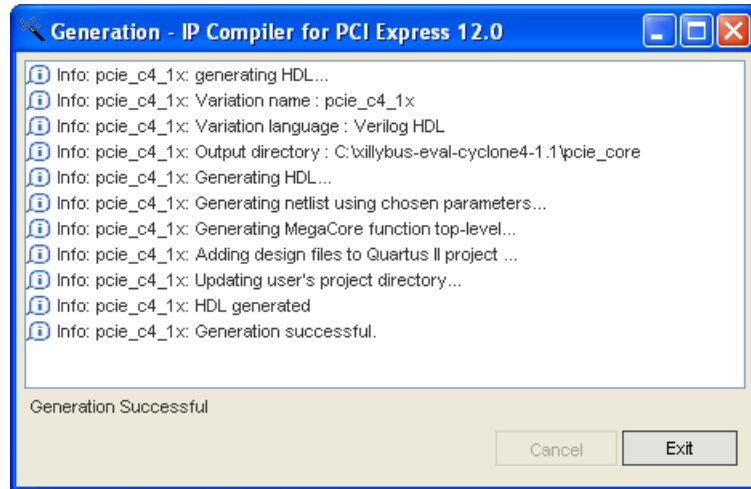


Megawizard가 “Specify a valid MegaWizard-generated variation file”라고 말하면서 파일 열기를 거부하면 이 문제에 대한 몇 가지 가능성이 있습니다.

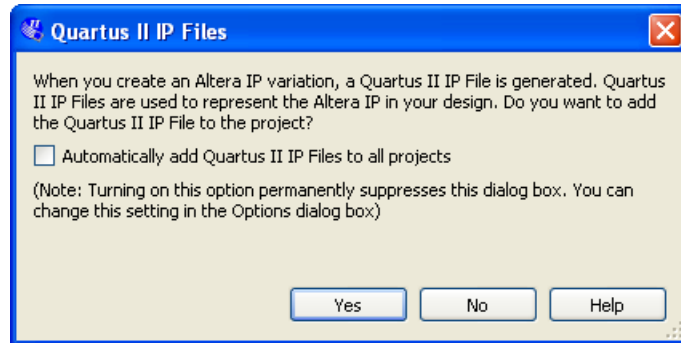
- 최신 FPGA(예: Cyclone V)로 작업하고 있습니다. 이 경우 PCIe 블록을 전혀 구축할 필요가 없습니다.
- Quartus 버전 번호를 현재 버전으로 변경하기 위해 변형 파일이 제대로 편집되지 않았습니다. 이 섹션의 시작 부분을 참조하십시오.
- Megawizard가 유효하지 않은 파일을 거부하는 경우 파일을 편집하고 다시 로드하려고 해도 도움이 되지 않습니다. Megawizard 프로그램을 종료하고 prompt 명령에서 다시 호출해야 합니다. 그렇지 않으면 파일이 제대로 편집된 경우에도 계속 거부됩니다.

열리는 창과 표시되는 매개변수는 FPGA 제품군마다 다릅니다(특히 “IP Compiler for PCI Express”와 다른 제목은 권장음).

변경 사항이 없어야 합니다. “Finish”를 클릭하기만 하면 됩니다. 다음과 같은 창에 진행 상황이 표시됩니다(최종 상태가 여기에 표시됨).



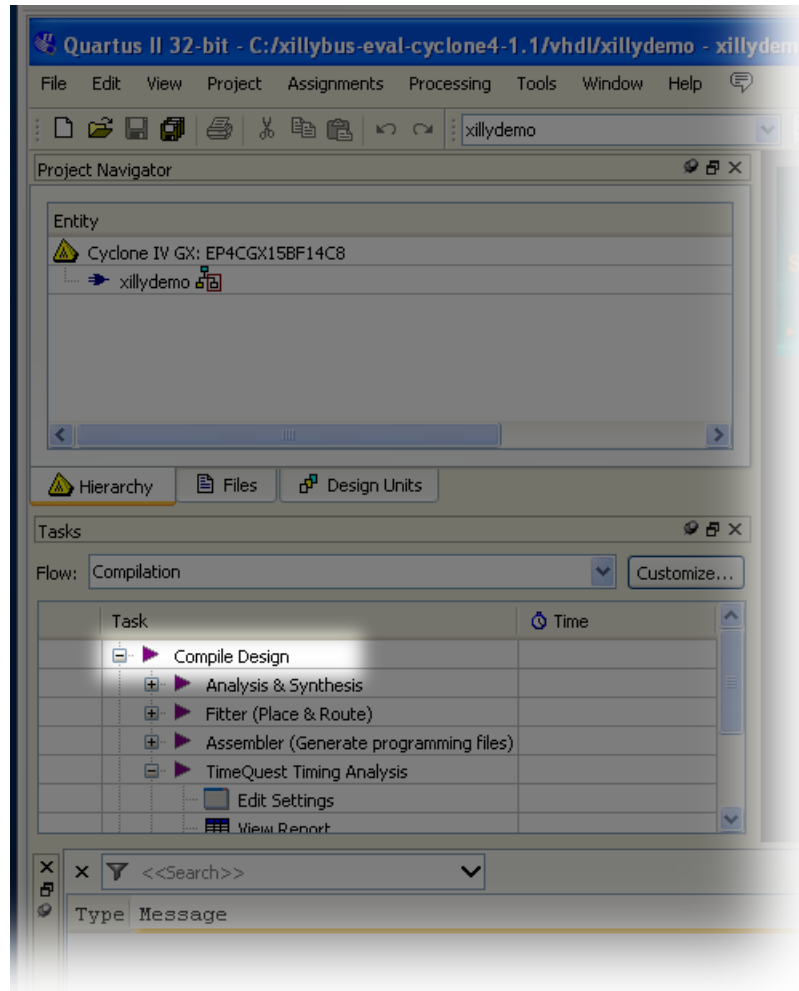
완료되면 종료를 클릭합니다. 프로젝트에 Quartus IP (QIP) 파일을 추가하도록 제안하는 다음 창이 즉시 나타날 수 있습니다.



QIP 파일을 프로젝트에 추가하면 어쨌든 무시되는 constraints가 포함된 불필요한 SDC 파일이 간접적으로 추가되기 때문에 선호하는 대답은 “No”입니다. 프로젝트에 이 QIP 파일이 있는 것은 무해하지만 compilation 중에 warnings가 많이 발생합니다. 일반적으로 SDC 파일에서 무시되는 constraint 각각에 대해 두 개의 warnings가 발생합니다.

3.3.3 design 파일의 Compilation

Quartus의 메인 창에서 “Compile Design”을 클릭하여 FPGA bitstream 파일을 생성합니다. 일부 Quartus 개정판에서는 compilation의 기본 절차가 “Rapid Recompile”로 설정되어 작업 창에서 “Rapid Recompile”만 허용할 수 있습니다. 이 경우 “Compilation”로 변경하고 진행합니다.



이 프로세스는 20-50 warnings(QIP 파일이 프로젝트에 포함되었는지 여부에 따라 다름)를 생성하지만 “Full Compilation was successful”라는 대화 상자로 끝나야 합니다.

오류나 Critical Warnings가 생성되지 않았는지 확인하는 것은 필수입니다(기본 Quartus 창 하단의 탭은 각 유형에 대한 메시지 수를 나타냄).

프로세스가 끝나면 프로그래밍 파일은 xillydemo.sof로 찾을 수 있습니다.

3.4 FPGA 프로그래밍

초기 개발 단계에서는 JTAG을 통해 FPGA를 로드하는 것이 좋습니다. 이것은 일반적으로 USB Blaster / Altera Download Cable(온보드 또는 외부)로 수행됩니다. JTAG을 통해 FPGA를 로드하는 방법에 대한 보드 지침을 참조하십시오.

XillyUSB 프로젝트의 경우 USB 인터페이스가 작동 중인 컴퓨터에 연결되어 있는 동안에도 FPGA를 언제든지 로드 및 다시 로드할 수 있습니다.

PCIe 프로젝트의 경우 컴퓨터의 전원을 켜기 전에 FPGA에 bitfile을 로드해야 합니다. 컴퓨터는 전원이 켜질 때 PCIe 주변기가 적절한 상태에 있을 것으로 예상하고 이후에 어떤 놀라움도 용납하지 않을 수 있습니다.

따라서 host가 실행 중인 동안에는 FPGA를 다시 로드하지 **마십시오**. PCIe 사양에는 hotplugging에 대한 지원이 필요하지만 마더보드는 일반적으로 PCIe 카드가 사라졌다가 다시 나타날 것으로 예상하지 않습니다. 따라서 일부 마더보드는 제대로 응답하지 않을 수 있습니다. 그럼에도 불구하고 운영 체제가 실행되는 동안 FPGA를 다시 로드하면 일부 마더보드에서 작동합니다.

Xillybus의 driver는 hotplugging에 제대로 응답하도록 설계되었지만 컴퓨터의 일반적인 안정성을 보장하는 것은 없습니다. 이것은 이 페이지에서 논의됩니다.

<https://xillybus.com/doc/hot-reconfiguration>

FPGA의 전원이 켜지고 컴퓨터의 전원을 켜면서 flash memory에서 로드되는 경우 FPGA가 충분히 빨리 로드되어 BIOS가 bus를 스캔할 때 PCIe 장치가 존재하도록 하는 것이 중요합니다.

4

수정

4.1 맞춤형 logic와 통합

Xillybus demo bundle은 application logic와 쉽게 통합되도록 구성되었습니다. 데이터를 연결하는 위치는 xillydemo.v 또는 xillydemo.vhd 파일입니다(선호하는 언어에 따라 다름). 번들의 다른 모든 HDL 파일은 host(Linux 또는 Windows)와 FPGA 간에 데이터를 전송하기 위해 Xillybus IP core를 사용할 목적으로 무시할 수 있습니다.

사용자 지정 logic designs가 있는 추가 HDL 파일은 3.3 단락에 설명된 대로 준비된 프로젝트에 추가한 다음 “Compile Design”을 클릭하여 다시 빌드할 수 있습니다. 초기 배포의 다른 단계를 반복할 필요가 없으므로 logic의 개발 주기는 상당히 빠르고 간단합니다.

Xillybus IP core를 사용자 지정 application logic에 연결할 때 FIFOs를 통해서만 Xillybus IP core와 상호 작용하고 logic로 동작을 모방하려고 시도하지 않는 것이 좋습니다. 최소한 첫 번째 단계에서는 아닙니다.

이에 대한 예외는 메모리 또는 register arrays를 Xillybus에 연결할 때이며, 이 경우 xillydemo 모듈에 표시된 예를 따라야 합니다.

xillydemo 모듈에서 FIFOs는 data loopback을 수행하는 데 사용됩니다. 즉, host에서 도착한 데이터가 다시 host로 전송됩니다. FIFO의 양쪽이 모두 Xillybus IP core에 연결되어 있으므로 core는 데이터 소스이자 데이터 소비자입니다.

실제 사용 시나리오에서는 FIFO의 측면 중 하나만 Xillybus IP core에 연결됩니다. FIFO의 다른 쪽은 데이터를 공급하거나 소비하는 application logic에 연결됩니다.

xillydemo 모듈에 사용되는 FIFOs는 양쪽이 Xillybus의 메인 clock에 의해 구동되기 때문에 양쪽(scfifo)에 대해 하나의 공통 clock에서만 작동합니다. 실제 애플리케이션에서는 읽기 및 쓰기용으로 별도의 clocks가 있는 FIFOs로 교체하는 것이 바람직할 수 있습니다. 이를 통해 bus_clk이 아닌 clock로 데이터 소스와 데이터 소비자를 구동할 수 있습니다. 이렇게 함으로써 FIFOs는 중재자 역할을 할 뿐만 아니라 적절한 clock domain 교차점 역할

도 합니다.

Xillybus IP core는 FPGA에서 host로의 streams에 대해 일반 FIFO 인터페이스("show-ahead"와 반대)를 예상합니다.

다음 문서는 맞춤형 logic 통합과 관련이 있습니다.

- logic design용 API: [Xillybus FPGA designer's guide](#)
- [Getting started with Xillybus on a Linux host](#)
- [Getting started with Xillybus on a Windows host](#)
- [Xillybus host application programming guide for Linux](#)
- [Xillybus host application programming guide for Windows](#)
- [The guide to defining a custom Xillybus IP core](#)

4.2 Altera의 PCIe IP / Megafunction에서 변경하기

절대적으로 필요한 경우가 아니면 Altera IP compiler for PCI Express Megafunction의 구성을 변경해서는 안 됩니다.

특히 receive buffers 할당에 대한 민감도가 높다. Xillybus IP core는 demo bundle에서 IP Compiler / MegaWizard가 제공하는 숫자에 따라 buffers의 크기에 의존합니다. PCIe core가 Xillybus IP core에서 예상한 것보다 적은 buffer 공간을 할당하는 경우 host에서 FPGA로의 통신에서 산발적인 데이터 오류가 발생할 수 있습니다. 이로 인해 이 방향에서 통신이 완전히 중단될 수도 있습니다. 이러한 문제는 이러한 buffers의 overflow를 유발하는 패킷이 FPGA에 도착한 결과입니다.

IP / Megafunction에 변경이 필요한 경우 Xillybus 웹 사이트에 게시된 이메일 주소를 통해 도움을 요청하십시오.

4.3 사용자 지정 프로젝트에 포함

원하는 경우 기존 Quartus 프로젝트에 Xillybus IP core를 포함하거나 처음부터 새 프로젝트를 생성할 수 있습니다. 이 섹션은 PCIe용 Xillybus에 관한 것이지만 XillyUSB에도 유사한 절차가 적용됩니다.

프로젝트가 이미 존재하지 않는 경우 새 프로젝트를 시작하고 원하는 HDL 언어 및 의도한 FPGA를 기반으로 설정합니다.

프로젝트에 Xillybus IP core를 포함하려면:

- `pcie_core/` 하위 디렉토리에서 프로젝트와 관련된 디렉토리로 `pcie_c4_1x.v` 파일(또는 유사 파일)을 복사합니다.
- Cyclone V, Arria V, Stratix V 및 series-10 FPGAs의 경우 `pcie_core/pcie_reconfig.qsys`(또는 `pcie_core/pcie_ip.ip`)도 동일한 디렉토리에 복사합니다.
- Altera의 PCIe 블록을 구축하기 위해 선택한 FPGA에 필요한 경우 3.3.2에 설명된 절차를 따르십시오.
- Cyclone IV용으로 다음 파일을 추가해야 합니다(복사본 가능).
 - `pcie_c4_1x.v`
 - `pcie_c4_1x_core.v`
 - `pcie_c4_1x_serdes.v`
 - `pcie_c4_1x_examples/chaining_dma/pcie_c4_1x_rs_hip.v`
 - `ip_compiler_for_pci_express-library/altpcie_hip_pipen1b.v`
 - `ip_compiler_for_pci_express-library/altpcie_reconfig_3cgx.v`
 - `ip_compiler_for_pci_express-library/altpcie_rs_serdes.v`

Stratix IV 및 Arria II의 경우 다음 파일을 추가해야 합니다.

- `pcie_s4_4x.v`
- `pcie_s4_4x_core.v`
- `pcie_s4_4x_serdes.v`
- `pcie_s4_4x_examples/chaining_dma/pcie_s4_4x_rs_hip.v`
- `ip_compiler_for_pci_express-library/altpcie_hip_pipen1b.v`
- `ip_compiler_for_pci_express-library/altpcie_reconfig_4sgx.v`
- `ip_compiler_for_pci_express-library/altpcie_rs_serdes.v`

series-V FPGAs의 경우 Qsys에 의해 생성되고 포함된 유사한 파일을 복사할 필요가 없습니다.

- 두 개의 `src/` 하위 디렉토리(언어 기본 설정에 따라 다름) 중 하나에 두 개의 HDL 파일 `xillybus.v` 및 `xillydemo.v(hd)`를 복사하고 프로젝트에 추가합니다.
- `src/` 디렉토리 중 하나의 SDC 파일에서 `constraints`를 채택하십시오(프로젝트의 top level 신호 이름과 일치시키기 위해 약간의 수정이 필요할 수 있음).
- `core/` 디렉터리에서 `xillybus_core.qxp` 또는 `xillybus_core.vqm`을 복사하고 이 파일을 프로젝트에도 추가합니다.

- xillydemo 모듈이 프로젝트의 top level module이 아닌 경우 해당 포트를 top level에 연결합니다.
- Xillybus IP core를 맞춤형 application logic에 부착하려면 기존 application logic을 원하는 모듈로 교체하여 xillydemo 모듈을 편집하십시오.

4.4 다른 보드 사용

4.4.1 일반적인

demo bundles 목록에 나타나지 않는 보드로 작업할 때 번들에서 약간의 수정이 필요합니다.

4.4.2 장치 제품군

demo bundle에는 특정 장치 제품군용 synthesized 파일인 QXP 또는 VQM 파일이 포함되어 있습니다. 이 제품군의 모든 장치와 함께 사용할 수 있습니다.

4.4.3 Pin placements

구입한 대부분의 보드에는 FPGA design의 자체 예가 있으며 해당 보드에서 PCIe 인터페이스가 사용되는 방식을 보여줍니다. 의도한 보드의 QSF 파일에서 관련 핀 할당을 찾고 핀 이름을 Xillybus의 QSF 파일에서 사용되는 이름으로 수정하는 것이 가장 쉬운 경우가 많습니다. 이렇게 하면 Xillybus의 프로젝트에서 사용되는 QSF 파일의 관련 행을 교체할 수 있습니다.

기능적 의미가 없는 4개의 user_led 와이어도 있습니다. 그러나 보드에 빈 LEDs가 있는 경우 통신 상태에 대한 몇 가지 표시를 제공하므로 연결하는 것이 좋습니다. user_led 신호는 active low logic을 가정합니다(logic '0'은 LED가 켜져 있음을 의미함).

4.4.4 PCIe만 해당: Clocking

xillydemo 모듈은 이 세 가지 clocks 중 일부 또는 전부를 입력으로 필요로 합니다. 필요한 clocks는 xillydemo 모듈에 대한 입력인 clocks입니다.

- pcie_refclk- host의 마더보드에서 제공하는 reference clock에 직접 연결됩니다. 이 clock의 주파수는 100 MHz여야 합니다. 이 clock은 host의 전원이 꺼져 있을 때와 PCIe 사양에서 허용하는 다른 상황에서 활성화되지 않을 수 있습니다. FPGA에서 요구하는 대로 마더보드의 clock와 FPGA에 연결된 clock 사이에 clock 클리닝 회로가 있을 수 있습니다.

이 입력 포트에 다른 clock 주파수가 제공되면(예: clock 클리너에서 제공하는 125 MHz) PCI Express 설정에 대해 IP Compiler에서 Xcvr_ref_clk 매개변수를 적절하게 변경해야 합니다. 이것은 단락 3.3에 표시된 대로 관련 IP Compiler / MegaWizard plug-in manager를 호출할 때 수행됩니다.

- clk_50– gigabit transceiver의 reconfig_clk을 구동하며 항상 활성 상태여야 합니다. 따라서 마더보드의 reference clock에 의존해서는 안 됩니다. 이 clock의 허용 주파수 범위는 FPGA 제품군에 따라 다릅니다(예: Cyclone IV의 37.5 MHz ~ 50 MHz). 이 clock의 사양은 관련 FPGA의 device handbook에서 “reconfig_clk”을 검색하여 찾을 수 있습니다.
- clk_125– transceiver에 필요한 상시 활성 clock을 구동합니다. 이 clock의 주파수는 125 MHz여야 하며 마더보드의 reference clock에 의존하지 않아야 합니다.

중요한:

pcie_refclk은 보드의 일부 오실레이터에 의해 구동되어서는 안 됩니다. host의 clock와 약간의 주파수 차이로 인해 통신이 불안정하거나 FPGA를 PCIe 장치로 감지하지 못합니다.

Cyclone IV GX Transceiver Starter Board용 demo bundle에서 clk_50 및 clk_125 입력은 보드에서 각각 50 MHz 및 125 MHz의 clock 소스에 연결된 I/O 핀에 의해 직접 구동됩니다. 보드에서 직접 적합한 clocks가 없는 경우 Altera에서 게시한 IP Compiler for PCI Express User Guide의 섹션 7에 설명된 대로 PLL을 사용하여 두 가지 모두를 생성할 수 있습니다.

PLL을 사용하는 경우 xillybus.v를 편집하여 reconfig_clk_locked 신호를 PLL의 잠금 표시기에 연결해야 합니다. 이것은 Stratix IV용 demo bundle에서 이미 수행되었습니다. DE4 보드는 125 MHz 주파수가 있는 항상 활성 clock을 제공하지 않기 때문입니다.

PCI Express용 PCI Compiler 파일이 생성된 후 pcie_core 서브디렉토리에 있는 파일 중 PLL을 사용한 예가 있습니다. 이것은 pcie_c4_1x_examples/chaining_dma 디렉토리에 pcie_c4_1x_example_chaining_pipen1b.v로 찾을 수 있습니다(또는 Cyclone IV 이외의 제품군으로 작업할 때 유사한 파일).

4.4.5 XillyUSB로 작업하기

XillyUSB는 SFP+ 인터페이스가 있는 다른 보드에서 사용할 수 있습니다. 이 경우 SFP+ 커넥터에 연결된 MGT를 사용하도록 design의 constraints를 설정하기만 하면 됩니다.

보드는 또한 MGT에 대해 낮은 jitter와 함께 125 MHz reference clock을 공급해야 합니다. USB 사양의 요구 사항에도 불구하고 Spread Spectrum Clocking (SSC)는 활성화되어서

는 안 됩니다(해당 옵션이 있는 경우): SSC reference clock이 사용되는 경우 MGT는 수신된 신호에 대해 제대로 잠기지 않습니다.

맞춤형 보드의 경우 SFP+ 커넥터의 핀이 FPGA의 MGT에 직접 연결되므로 sfp2usb 모듈의 회로도 참조하는 것이 좋습니다. sfp2usb 모듈에서와 같이 SSRX 와이어를 교체하는 것은 선택 사항입니다. 이는 PCB design을 단순화하는 경우에만 권장됩니다.

원하는 경우 SSTX 와이어를 교체하는 것도 가능합니다. 이를 위해서는 전송된 비트의 극성이 반전되도록 *_frontend.v 파일을 편집해야 하므로 전선 쌍 교환을 보상합니다. USB 사양은 극성 교체로도 link partners가 제대로 작동해야 하므로 이 편집 없이도 USB 연결이 제대로 작동할 가능성이 높습니다. 그러나 이것에 의존하지 않는 것이 좋습니다.

5

문제 해결

5.1 implementation 중 오류

때로는 Altera의 도구 릴리스 간에 약간의 차이가 있습니다. 이로 인해 SOF 파일 생성을 위한 implementation 실행이 실패할 수 있습니다.

문제가 신속하게 해결되지 않으면 Xillybus 웹 사이트에 제공된 이메일 주소를 통해 도움을 요청하십시오. 특히 도구에서 보고된 첫 번째 오류 주변에서 실패한 프로세스의 출력 로그를 첨부하십시오. 또한 design에서 사용자 지정 변경이 이루어진 경우(예: demo bundle에서 전환) 이러한 변경 사항을 자세히 설명하십시오. 또한 어떤 버전의 Quartus 도구가 사용되었는지 명시하십시오.

5.2 PCIe 하드웨어 문제

일반적으로 PCIe 카드는 host의 BIOS 및/또는 운영 체제에서 올바르게 감지되고 host의 driver가 성공적으로 실행됩니다.

대부분의 PC 컴퓨터에서 BIOS는 boot 프로세스 초기에 감지된 주변 장치 목록을 간략하게 표시합니다. Xillybus 인터페이스가 성공적으로 감지되면 vendor ID 1172 및 device ID EBEB가 있는 주변 장치가 목록에 나타납니다.

운영 체제의 카드 감지에 대해서는 다음 두 문서 중 해당하는 문서를 참조하십시오.

- [Getting started with Xillybus on a Linux host](#)
- [Getting started with Xillybus on a Windows host](#)

카드 감지 실패(또는 컴퓨터의 boot 프로세스 실패)는 Xillybus IP core와 관련이 없으며 Altera의 PCIe IP core를 사용하여 bus와 연결합니다.

처음에는 다음을 확인하는 것이 좋습니다.

- bitstream은 컴퓨터의 전원이 켜졌을 때(또는 PCI-SIG 사양 측면에서 전원이 켜진 직후) 이미 FPGA에 로드되었습니다.
- reference clock을 포함하여 PCIe 와이어의 pinouts가 정확합니다(이는 fitter의 보고서에서 확인할 수 있음).
- 보드는 올바른 reference clock을 FPGA에 공급합니다.

문제가 즉시 발견되지 않으면 보드와 함께 제공된 PCIe용 샘플 프로젝트를 시도하는 것이 좋습니다. 잘못된 점퍼 설정과 하드웨어 결함이 나타날 수 있습니다.

이 샘플에서는 카드가 감지되지만 Xillybus에서는 감지되지 않는 경우 두 designs의 pinouts를 비교하는 것이 도움이 될 수 있습니다. 동일한 경우 다음 단계는 관련 GUI 도구를 각각 호출하여 Altera의 PCIe cores 속성을 비교하는 것입니다.

다음 구성 요소를 조정해야 할 수 있습니다.

- reference clock의 주파수.
- base class 및 sub class(가능성은 없지만 class가 알 수 없는 것으로 간주되는 경우 상대적으로 오래된 일부 PC 컴퓨터에서 boot 프로세스가 실패했습니다).
- 기본 주소 register 설정, vendor ID, device ID 및 인터럽트 설정을 제외하고 다르게 구성된 기타 속성은 변경되어서는 안 됩니다.

문제가 지속되면 Xillybus 웹 사이트에 제공된 이메일 주소를 통해 도움을 요청하십시오.