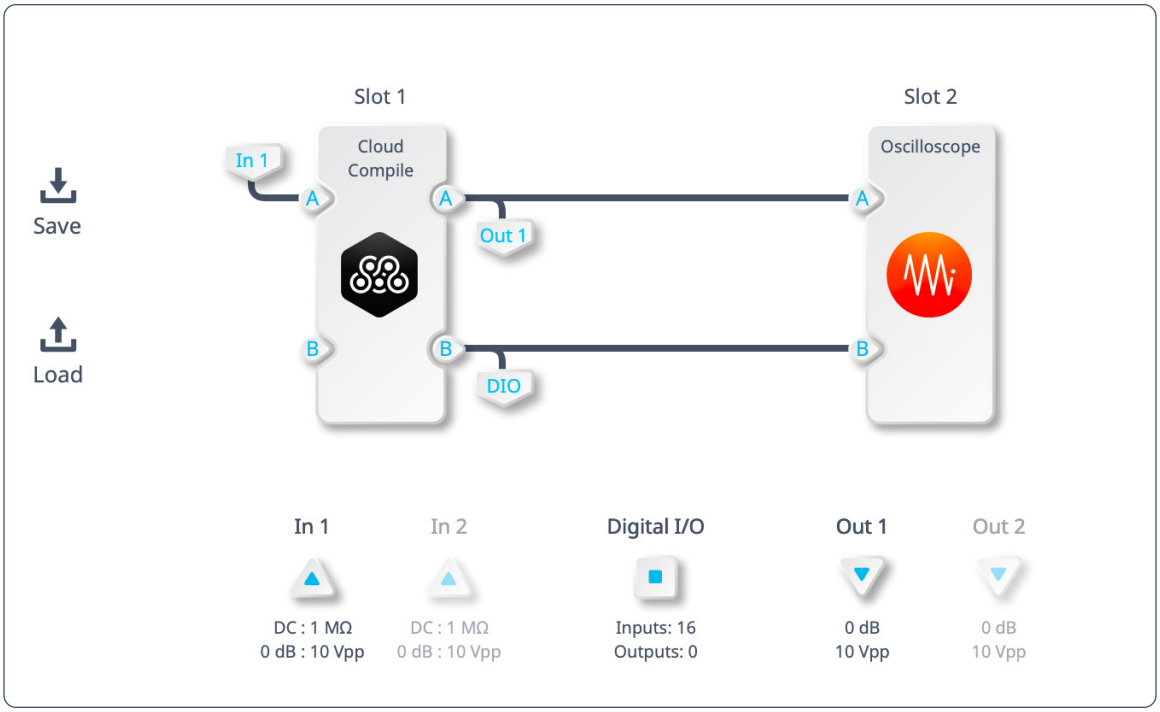
Moku Cloud Compile

Available on Moku:Go and Moku:Pro

Moku Cloud Compile (MCC) enables users to deploy custom code directly on Moku hardware, opening up endless new possibilities. This allows for unprecedented customization capabilities with minimal development time. A custom wrapper is available to ensure Moku Cloud Compile creations can interface directly with the frontend of compatible hardware. Custom designs can run alongside any supported instrument in the Moku suite in Multi-instrument Mode. By opening access to the FPGA that powers Moku hardware, new features or functions can be deployed in minutes. The cloud-based workflow greatly simplifies setup and deployment.



Example use cases

- Create custom math functions, gain or offset and create more complex signal processing flows
- Design and deploy a custom instrument like a boxcar averager, specific DC sequencing, or more advanced cross correlation functions
- Create custom pulse or function sequences
- Digitally sum, subtract, or otherwise combine signals, or create gaussian noise for testing
- Combine with MATLAB HDL coder to produce noise generators, random sequences, or square root functions

```
14
15  library IEEE;
16  use IEEE.Std_Logic_1164.all;
17  use IEEE.Numeric_Std.all;
18
19  library Moku;
20  use Moku.Support.sum_no_overflow;
21
22  architecture Behavioural of CustomWrapper is
23
24      signal t1 : unsigned(39 downto 0);
25      signal t2 : unsigned(31 downto 0);
26      signal t3 : unsigned(31 downto 0);
27      signal t4 : unsigned(31 downto 0);
28
29      signal HI_LVL : signed(15 downto 0);
30      signal LO_LVL : signed(15 downto 0);
31      signal Threshold : signed(15 downto 0);
32
```

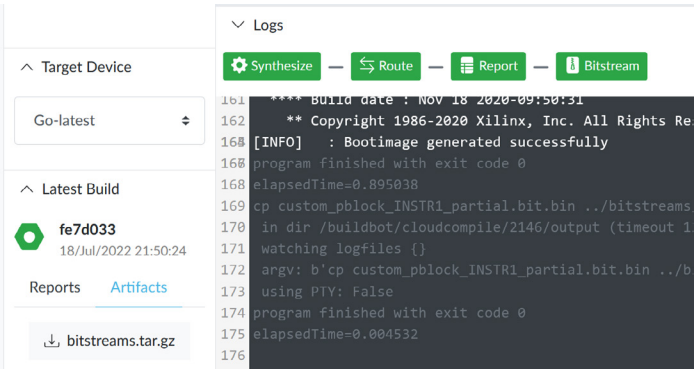
Get started with free example code available on https://compile.liquidinstruments.com/docs/

Powerful customization made easy

Multi-instrument Mode allows Moku Cloud Compile to interact with Moku's suite of standard instruments. For example, the Oscilloscope can be used to observe the custom code outputs, or use the custom code to drive modulation onto a Waveform Generator signal. By harnessing the power of the existing instrument suite, users can create, compile, and validate their designs on one hardware platform.

Register	Decimal (Unsigned)	Decimal (Signed)	Hexadecimal	Binary
Control0	1,234	1,234	0000 04D2	0000 0000 0000 0000 0000 0100 1101 0010
Control1	2,882,371,583	-1,412,595,713	ABCD 7FFF	1010 1011 1100 1101 0111 1111 1111 1111
Control2	33,596,509	33,596,509	0200 A45D	0000 0010 0000 0000 1010 0100 0101 1101

Custom code can be controlled via app-based registers to adjust gains or change operating modes in real-time. Register changes are reflected immediately in the running VHDL for quick adjustment of design parameters.



Learn, explore, deploy

Compile, synthesize, and route entirely in the cloud. Users can code in VDHL or use third-party code generators like HDL Coder from Mathworks.

The synthesized design forms a bitstream file that is hardware deployable via a simple web-based interface. Cloud based synthesis allows users to avoid the need to install, debug, and maintain complex local tools.

Available on Moku:Go and Moku:Pro

Moku Cloud Compile is intended to lower the intimidating barrier of hardware programming by creating a platform accessible to everyone, from those just starting out programming FPGAs to experts.

Moku:Go

Students can learn VHDL without complex software installations. Deploy Moku Cloud Compile creations alongside another instrument in Multi-instrument Mode to test designs in realtime.

Moku:Pro

Customize advanced test setups by deploying custom DSP alongside up to three additional instruments in Moku:Pro's 4 slot Multi-instrument Mode, enabling you to quickly tailor your test equipment to your requirements.

	Moku:Go	Moku:Pro
FPGA	Zync 7000	Ultrascale+
Core clock	31.25 MHz	312.5 MHz
Sample rate	31.25/125 MSa/s	312.5/1250 MSa/s
Look-up tables	20,000	48,400
Flip-flops	40,000	96,800
Block RAM (36k)	50	154
DSP	100	432

All specifications are per slot