

PYNQ-Z1 Board Reference Manual

Revised April 13, 2017

Overview

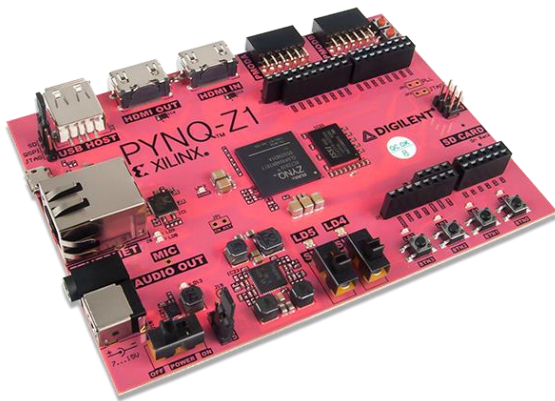
The PYNQ-Z1 board is designed to be used with PYNQ, a new open-source framework that enables embedded programmers to exploit the capabilities of Xilinx Zynq All Programmable SoCs (APSoCs) without having to design programmable logic circuits. Instead the APSoC is programmed using Python, with the code developed and tested directly on the PYNQ-Z1. The programmable logic circuits are imported as hardware libraries and programmed through their APIs in essentially the same way that the software libraries are imported and programmed.

The PYNQ-Z1 board is the hardware platform for the PYNQ open-source framework. The software running on the ARM A9 CPUs includes:

- A web server hosting the Jupyter Notebook design environment
- The IPython kernel and packages
- Linux
- Base hardware library and API for the FPGA

For designers who want to extend the base system by contributing new hardware libraries, Xilinx Vivado WebPACK tools are available free of cost.

To find out more about PYNQ, please see the project webpage at www.pynq.io. Here you will find materials to help you get started and a forum for contacting the supporting community.



The PYNQ-Z1.

- **ZYNQ XC7Z020-1CLG400C**
 - 650MHz dual-core Cortex-A9 processor
 - DDR3 memory controller with 8 DMA channels and 4 High Performance AXI3 Slave ports
 - High-bandwidth peripheral controllers: 1G Ethernet, USB 2.0, SDIO
 - Low-bandwidth peripheral controller: SPI, UART, CAN, I2C
 - Programmable from JTAG, Quad-SPI flash, and microSD card
 - Programmable logic equivalent to Artix-7 FPGA
 - 13,300 logic slices, each with four 6-input LUTs and 8 flip-flops
 - 630 KB of fast block RAM
 - 4 clock management tiles, each with a phase-locked loop (PLL) and mixed-mode clock manager (MMCM)
 - 220 DSP slices
 - On-chip analog-to-digital converter (XADC)
- **Memory**
 - 512MB DDR3 with 16-bit bus @ 1050Mbps
 - 16MB Quad-SPI Flash with factory programmed 48-bit globally unique EUJ-48/64™ compatible identifier
 - microSD slot
- **USB and Ethernet**
 - Gigabit Ethernet PHY
 - USB-JTAG Programming circuitry
 - USB-UART bridge
 - USB OTG PHY (supports host only)
- **Audio and Video**
 - HDMI sink port (input)
 - HDMI source port (output)
 - Microphone with PDM interface
 - PWM driven mono audio output with 3.5mm jack
- **Switches, Push-buttons, and LEDs**
 - 4 push-buttons
 - 2 slide switches
 - 4 LEDs
 - 2 RGB LEDs
- **Expansion Connectors**
 - Two standard Pmod ports
 - 16 Total FPGA I/O
 - Arduino/chipKIT Shield connector
 - 49 Total FPGA I/O
 - 6 Single-ended 0-3.3V Analog inputs to XADC
 - 4 Differential 0-1.0V Analog inputs to XADC
- **Power**
 - Powered from USB or any 7V-15V external power source

The board can be purchased stand-alone or with an accessory kit that contains a 12V/3A power adapter, 10 foot Ethernet cable, USB A to Micro-B cable, and an 8GB, speed class 10 microSD card loaded with the PYNQ image is available. For more information on purchasing, see the [PYNQ Product Page](#).

1 Power Supplies

The PYNQ-Z1 can be powered from the Digilent USB-JTAG-UART port (J14) or from some other type of power source such as a battery or external power supply. Jumper JP5 (near the power switch) determines which power source is used.

A USB 2.0 port can deliver maximum 0.5A of current according to the specifications. This should provide enough power for lower complexity designs. More demanding applications, including any that drive multiple peripheral boards or other USB devices, might require more power than the USB port can provide. In this case, power consumption will increase until it's limited by the USB host. This limit varies a lot between manufacturers of host computers and depends on many factors. When in current limit, once the voltage rails dip below their nominal value, the Zynq is reset by the Power-on Reset signal and power consumption returns to its idle value. Also, some applications may need to run without being connected to a PC's USB port. In these instances an external power supply or battery can be used.

An external power supply (e.g. wall wart) can be used by plugging it into the power jack (J18) and setting jumper JP5 to "REG". The supply must use a coax, center-positive 2.1mm internal-diameter plug, and deliver 7VDC to 15VDC. Suitable supplies can be purchased from the Digilent website or through catalog vendors like DigiKey. Power supply voltages above 15VDC might cause permanent damage. A suitable external power supply is included with the PYNQ-Z1 accessory kit.

Similar to using an external power supply, a battery can be used to power the PYNQ-Z1 by attaching it to the shield connector and setting jumper JP5 to "REG". The positive terminal of the battery must be connected to the pin labeled "VIN" on J7, and the negative terminal must be connected to the pin labeled GND on J7.

The on-board Texas Instruments TPS65400 PMU creates the required 3.3V, 1.8V, 1.5V, and 1.0V supplies from the main power input. Table 1.1 provides additional information (typical currents depend strongly on Zynq configuration and the values provided are typical of medium size/speed designs).

All on-board power supplies are enabled or disabled by the power switch SW4. The power indicator LED (LD13) is on when all the supply rails reach their nominal voltage.

Supply	Circuits	Current (max/typical)
3.3V	FPGA I/O, USB ports, Clocks, Ethernet, SD slot, Flash, HDMI	1.6A/0.1A to 1.5A
1.0V	FPGA, Ethernet Core	2.6A/0.2A to 2.1A
1.5V	DDR3	1.8A/0.1A to 1.2A
1.8V	FPGA Auxiliary, Ethernet I/O, USB Controller	1.8A/0.1A to 0.6A

Table 1.1. PYNQ-Z1 power supplies.

2 Zynq APSoC Architecture

The Zynq APSoC is divided into two distinct subsystems: The Processing System (PS) and the Programmable Logic (PL). Figure 2.1 shows an overview of the Zynq APSoC architecture, with the PS colored light green and the PL in yellow. Note that the PCIe Gen2 controller and Multi-gigabit transceivers are not available on the Zynq-7020 device.

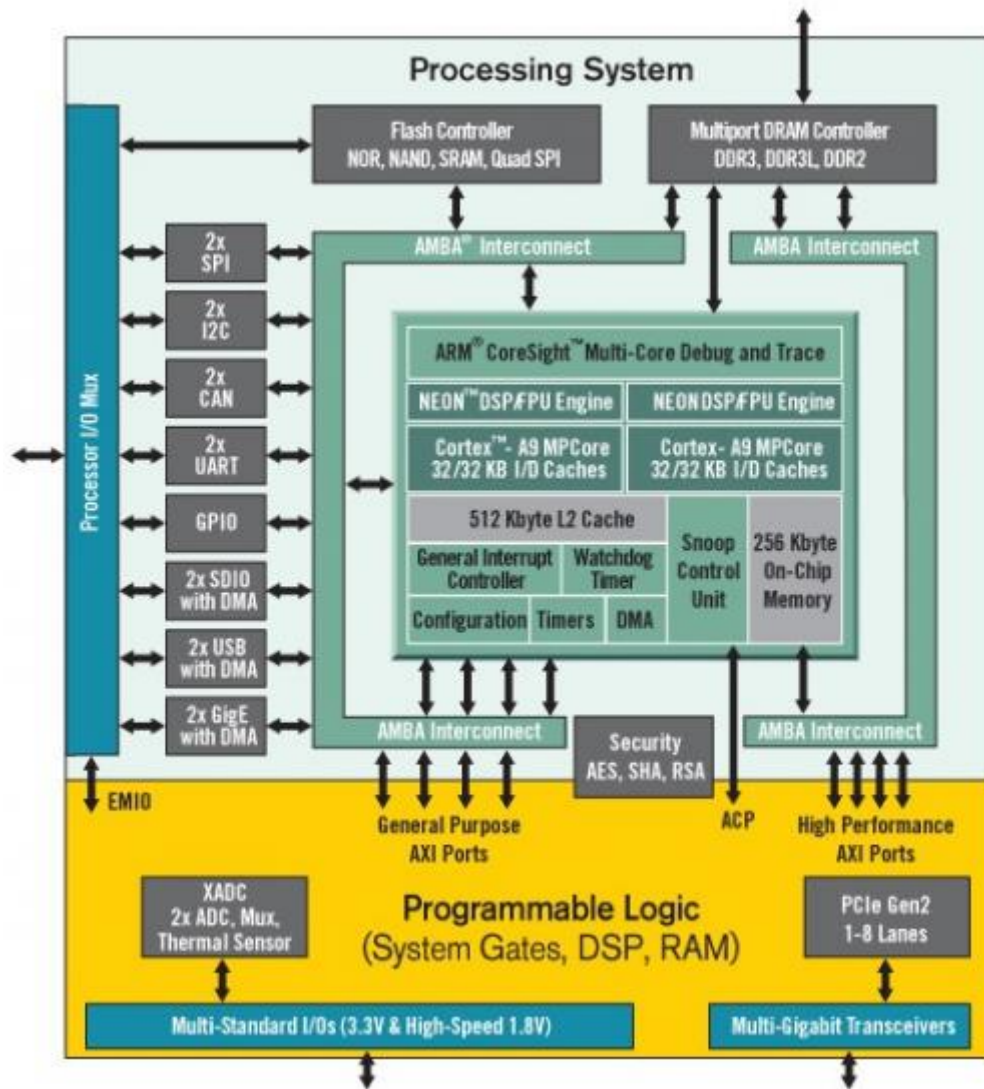


Figure 2.1. Zynq APSoC architecture.

The PL is nearly identical to a Xilinx 7-series Artix FPGA, except that it contains several dedicated ports and buses that tightly couple it to the PS. The PL also does not contain the same configuration hardware as a typical 7-series FPGA, and it must be configured either directly by the processor or via the JTAG port.

The PS consists of many components, including the Application Processing Unit (APU, which includes 2 Cortex-A9 processors), Advanced Microcontroller Bus Architecture (AMBA) Interconnect, DDR3 Memory controller, and various peripheral controllers with their inputs and outputs multiplexed to 54 dedicated pins (called Multiplexed I/O, or MIO pins). Peripheral controllers that do not have their inputs and outputs connected to MIO pins can instead route their I/O through the PL, via the Extended-MIO (EMIO) interface. The peripheral controllers are connected to the processors as slaves via the AMBA interconnect, and contain readable/writable control registers that are addressable in the processors' memory space. The programmable logic is also connected to the interconnect as a slave, and designs can implement multiple cores in the FPGA fabric that each also contain addressable control registers. Furthermore, cores implemented in the PL can trigger interrupts to the processors (connections not shown in Fig. 3) and perform DMA accesses to DDR3 memory.

There are many aspects of the Zynq APSoC architecture that are beyond the scope of this document. For a complete and thorough description, refer to the [Zynq Technical Reference manual](#).

Table 2.1 depicts the external components connected to the MIO pins of the PYNQ-Z1. The Zynq Presets File found on the [PYNQ-Z1 Resource Center](#) can be imported into EDK and Vivado Designs to properly configure the PS to work with these peripherals.

MIO 500 3.3 V	Peripherals				
Pin	ENET 0	SPI Flash	USB 0	Shield	UART 0
0 (N/C)					
1		CS			
2		DQ0			
3		DQ1			
4		DQ2			
5		DQ3			
6		SCLK			
7 (N/C)					
8		SLCK FB			
9	Ethernet Reset				
10	Ethernet Interrupt				
11			USB Over Current		
12				Shield Reset	
13 (N/C)					
14					UART Input
15					UART Output

MIO 501 1.8V	Peripherals		
Pin	ENET 0	USB 0	SDIO 0
16	TXCK		
17	TXD0		
18	TXD1		
19	TXD2		
20	TXD3		
21	TXCTL		
22	RXCK		
23	RXD0		
24	RXD1		
25	RXD2		
26	RXD3		
27	RXCTL		
28		DATA4	
29		DIR	
30		STP	
31		NXT	
32		DATA0	
33		DATA1	
34		DATA2	
35		DATA3	
36		CLK	
37		DATA5	
38		DATA6	

MIO 501 1.8V	Peripherals		
Pin	ENET 0	USB 0	SDIO 0
39		DATA7	
40			CCLK
41			CMD
42			D0
43			D1
44			D2
45			D3
46		RESETN	
47			CD
48 (N/C)			
49 (N/C)			
50 (N/C)			
51 (N/C)			
52	MDC		
53	MDIO		

Table 2.1. MIO Pinout.

3 Zynq Configuration

Unlike Xilinx FPGA devices, APSoC devices such as the Zynq-7020 are designed around the processor, which acts as a master to the programmable logic fabric and all other on-chip peripherals in the processing system. This causes the Zynq boot process to be more similar to that of a microcontroller than an FPGA. This process involves the processor loading and executing a Zynq Boot Image, which includes a First Stage Bootloader (FSBL), a bitstream for configuring the programmable logic (optional), and a user application. The boot process is broken into three stages:

Stage 0

After the PYNQ-Z1 is powered on or the Zynq is reset (in software or by pressing SRST), one of the processors (CPU0) begins executing an internal piece of read-only code called the BootROM. If and only if the Zynq was just powered on, the BootROM will first latch the state of the mode pins into the mode register (the mode pins are attached to JP4 on the PYNQ-Z1). If the BootROM is being executed due to a reset event, then the mode pins are not latched, and the previous state of the mode register is used. This means that the PYNQ-Z1 needs a power cycle to register any change in the programming mode jumper (JP4). Next, the BootROM copies an FSBL from the form of non-volatile memory specified by the mode register to the 256 KB of internal RAM within the APU (called On-Chip Memory, or OCM). The FSBL must be wrapped up in a Zynq Boot Image in order for the BootROM to properly copy it. The last thing the BootROM does is hand off execution to the FSBL in OCM.

Stage 1

During this stage, the FSBL first finishes configuring the PS components, such as the DDR memory controller. Then, if a bitstream is present in the Zynq Boot Image, it is read and used to configure the PL. Finally, the user application is loaded into memory from the Zynq Boot Image, and execution is handed off to it.

Stage 2

The last stage is the execution of the user application that was loaded by the FSBL. This can be any sort of program, from a simple “Hello World” design, to a Second Stage Boot loader used to boot an operating system like Linux. For a more thorough explanation of the boot process, refer to Chapter 6 of the [Zynq Technical Reference manual](#).

The Zynq Boot Image is created using Vivado and Xilinx Software Development Kit (Xilinx SDK). For information on creating this image please refer to the available Xilinx documentation for these tools.

The PYNQ-Z1 supports three different boot modes: microSD, Quad SPI Flash, and JTAG. The boot mode is selected using the Mode jumper (JP4), which affects the state of the Zynq configuration pins after power-on. Figure 3.1 depicts how the Zynq configuration pins are connected on the PYNQ-Z1.

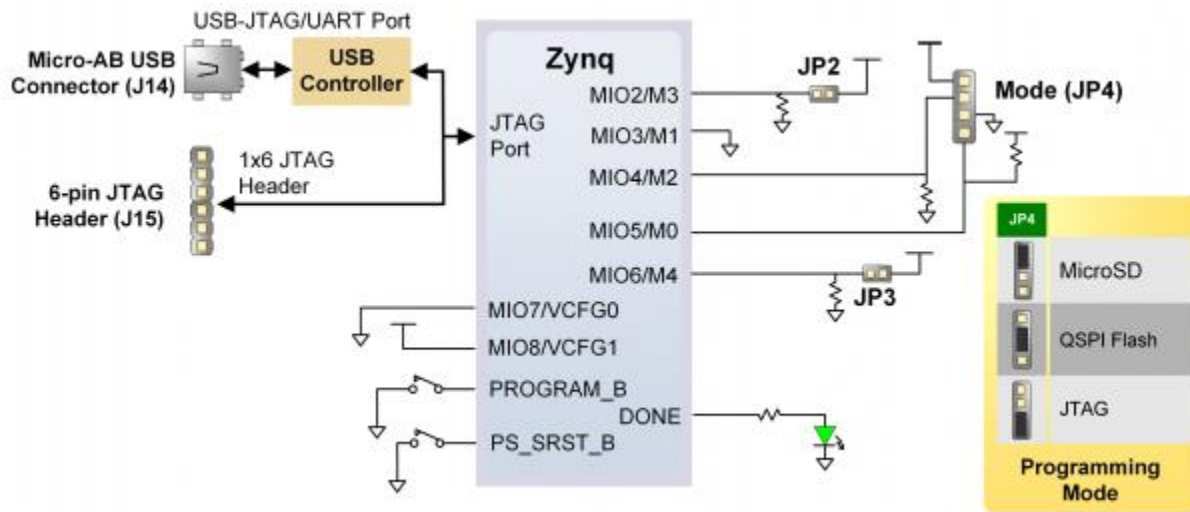


Figure 3.1. PYNQ-Z1 configuration pins.

The three boot modes are described in the following sections.

3.1 microSD Boot Mode

The PYNQ-Z1 supports booting from a microSD card inserted into connector J9. The following procedure will allow you to boot the Zynq from microSD with a standard Zynq Boot Image created with the Xilinx tools:

1. Format the microSD card with a FAT32 file system.
2. Copy the Zynq Boot Image created with Xilinx SDK to the microSD card.
3. Rename the Zynq Boot Image on the microSD card to BOOT.bin.
4. Eject the microSD card from your computer and insert it into connector J9 on the PYNQ-Z1.
5. Attach a power source to the PYNQ-Z1 and select it using JP5.
6. Place a single jumper on JP4, shorting the two top pins (labeled “SD”).
7. Turn the board on. The board will now boot the image on the microSD card.

In order to boot the PYNQ-Z1 into the PYNQ software environment, the microSD card must be formatted with a specially created disk image. Refer to www.pyngq.io for instructions on obtaining this image file and flashing it to a microSD card. Once the microSD card has been flashed with the image, the PYNQ-Z1 can be booted with it by following the instructions above starting at step 4.

3.2 Quad SPI Boot Mode

The PYNQ-Z1 has an onboard 16MB Quad-SPI Flash that the Zynq can boot from. Documentation available from Xilinx describes how to use Xilinx SDK to program a Zynq Boot Image into a Flash device attached to the Zynq. Once the Quad SPI Flash has been loaded with a Zynq Boot Image, the following steps can be followed to boot from it:

1. Attach a power source to the PYNQ-Z1 and select it using JP5.
2. Place a single jumper on JP4, shorting the two center pins (labeled “QSPI”).
3. Turn the board on. The board will now boot the image stored in the Quad SPI flash.

3.3 JTAG Boot Mode

When placed in JTAG boot mode, the processor will wait until software is loaded by a host computer using the Xilinx tools. After software has been loaded, it is possible to either let the software begin executing, or step through it line by line using Xilinx SDK.

It is also possible to directly configure the PL over JTAG, independent of the processor. This can be done using the Vivado Hardware Server.

The PYNQ-Z1 is configured to boot in Cascaded JTAG mode, which allows the PS to be accessed via the same JTAG port as the PL. It is also possible to boot the PYNQ-Z1 in Independent JTAG mode by loading a jumper in JP2 and shorting it. This will cause the PS to not be accessible from the onboard JTAG circuitry, and only the PL will be visible in the scan chain. To access the PS over JTAG while in independent JTAG mode, users will have to route the signals for the PJTAG peripheral over EMIO, and use an external device to communicate with it.

4 Quad SPI Flash

The PYNQ-Z1 features a Quad SPI serial NOR flash. The Spansion S25FL128S is used on this board. The Multi-I/O SPI Flash memory is used to provide non-volatile code and data storage. It can be used to initialize the PS subsystem as well as configure the PL subsystem.

The relevant device attributes are:

- 16 MB
- x1, x2, and x4 support
- Bus speeds up to 104 MHz, supporting Zynq configuration rates @ 100 MHz. In Quad SPI mode, this translates to 400Mbs
- Powered from 3.3V

The SPI Flash connects to the Zynq-7000 APSoC and supports the Quad SPI interface. This requires connection to specific pins in MIO Bank 0/500, specifically MIO[1:6,8] as outlined in the Zynq datasheet. Quad-SPI feedback mode is used, thus `qspi_sclk_fb_out/MIO[8]` is left to freely toggle and is connected only to a 20K pull-up resistor to 3.3V. This allows a Quad SPI clock frequency greater than FQSPICLK2 (See the [Zynq Technical Reference manual](#) for more on this).

5 DDR Memory

The PYNQ-Z1 includes an IS43TR16256A-125KBL DDR3 memory components creating a single rank, 16-bit wide interface and a total of 512MiB of capacity. The DDR3 is connected to the hard memory controller in the Processor Subsystem (PS), as outlined in the Zynq documentation.

The PS incorporates an AXI memory port interface, a DDR controller, the associated PHY, and a dedicated I/O bank. DDR3 memory interface speeds up to 533 MHz/1066 Mbps are supported.¹

PYNQ-Z1 was routed with 40 ohm (+/-10%) trace impedance for single-ended signals, and differential clock and strobes set to 80 ohms (+/-10%). A feature called DCI (Digitally Controlled Impedance) is used to match the drive strength and termination impedance of the PS pins to the trace impedance. On the memory side, each chip calibrates its on-die termination and drive strength using a 240-ohm resistor on the ZQ pin.

Due to layout reasons, the two data byte groups (DQ[0-7], DQ[8-15]) were swapped. To the same effect, the data bits inside byte groups were swapped as well. These changes are transparent to the user. During the whole design process the Xilinx PCB guidelines were followed.

Both the memory chips and the PS DDR bank are powered from the 1.5V supply. The mid-point reference of 0.75V is created with a simple resistor divider and is available to the Zynq as external reference.

For proper operation, it is essential that the PS memory controller is configured properly. Settings range from the actual memory flavor to the board trace delays. For your convenience, the Zynq presets file for the PYNQ-Z1 is provided on the [resource center](#) and automatically configures the Zynq Processing System IP core with the correct parameters.

For best DDR3 performance, DRAM training is enabled for write leveling, read gate, and read data eye options in the PS Configuration Tool in Xilinx tools. Training is done dynamically by the controller to account for board delays, process variations and thermal drift. Optimum starting values for the training process are the board delays (propagation delays) for certain memory signals.

Board delays are specified for each of the byte groups. These parameters are board-specific and were calculated from the PCB trace length reports. The DQS to CLK Delay and Board Delay values are calculated specific to the PYNQ-Z1 memory interface PCB design.

For more details on memory controller operation, refer to the Xilinx [Zynq Technical Reference manual](#).

6 USB UART Bridge (Serial Port)

The PYNQ-Z1 includes an FTDI FT2232HQ USB-UART bridge (attached to connector J14) that lets you use PC applications to communicate with the board using standard COM port commands (or the tty interface in Linux). Drivers are automatically installed in Windows and newer versions of Linux. Serial port data is exchanged with the Zynq using a two-wire serial port (TXD/RXD). After the drivers are installed, I/O commands can be used from the PC directed to the COM port to produce serial data traffic on the Zynq pins. The port is tied to PS (MIO) pins and can be used in combination with the UART 0 controller.

¹ Maximum actual clock frequency is 525 MHz on the PYNQ-Z1 due to PLL limitation.

The Zynq presets file (available in the [PYNQ-Z1 Resource Center](#)) takes care of mapping the correct MIO pins to the UART 0 controller and uses the following default protocol parameters: 115200 baud rate, 1 stop bit, no parity, 8-bit character length.

Two on-board status LEDs provide visual feedback on traffic flowing through the port: the transmit LED(LD11) and the receive LED (LD10). Signal names that imply direction are from the point-of-view of the DTE (Data Terminal Equipment), in this case the PC.

The FT2232HQ is also used as the controller for the Digilent USB-JTAG circuitry, but the USB-UART and USB-JTAG functions behave entirely independent of one another. Programmers interested in using the UART functionality of the FT2232 within their design do not need to worry about the JTAG circuitry interfering with the UART data transfers, and vice-versa. The combination of these two features into a single device allows the PYNQ-Z1 to be programmed, communicated with via UART, and powered from a computer attached with a single Micro USB cable.

The DTR signal from the UART controller on the FT2232HQ is connected to MIO12 of the Zynq device via JP1. Should the Arduino IDE be ported to work with the PYNQ-Z1, this jumper can be shorted and MIO12 could be used to place the PYNQ-Z1 in a “ready to receive a new sketch” state. This would mimic the behavior of typical Arduino IDE boot-loaders.

7 microSD Slot

The PYNQ-Z1 provides a microSD slot (J9) for non-volatile external memory storage as well as booting the Zynq. The slot is wired to Bank 1/501 MIO[40-47], including Card Detect. On the PS side peripheral SDIO 0 is mapped out to these pins and controls communication with the SD card. The pinout can be seen in Table 7.1. The peripheral controller supports 1-bit and 4-bit SD transfer modes, but does not support SPI mode. Based on the [Zynq Technical Reference manual](#), SDIO host mode is the only mode supported.

Signal Name	Description	Zynq Pin	SD Slot Pin
SD_D0	Data[0]	MIO42	7
SD_D1	Data[1]	MIO43	8
SD_D2	Data[2]	MIO44	1
SD_D3	Data[3]	MIO45	2
SD_CCLK	Clock	MIO40	5
SD_CMD	Command	MIO41	3
SD_CD	Card Detect	MIO47	9

Table 7.1. microSD pinout.

The SD slot is powered from 3.3V, but is connected through MIO Bank 1/501 (1.8V). Therefore, a TI TXS02612 level shifter performs this translation. The TXS02612 is actually 2-port SDIO port expander, but only its level shifter function is used. The connection diagram can be seen on Figure 7.1. Mapping out the correct pins and configuring the interface is handled by the PYNQ-Z1 Zynq presets file, available on the [PYNQ-Z1 Resource Center](#).

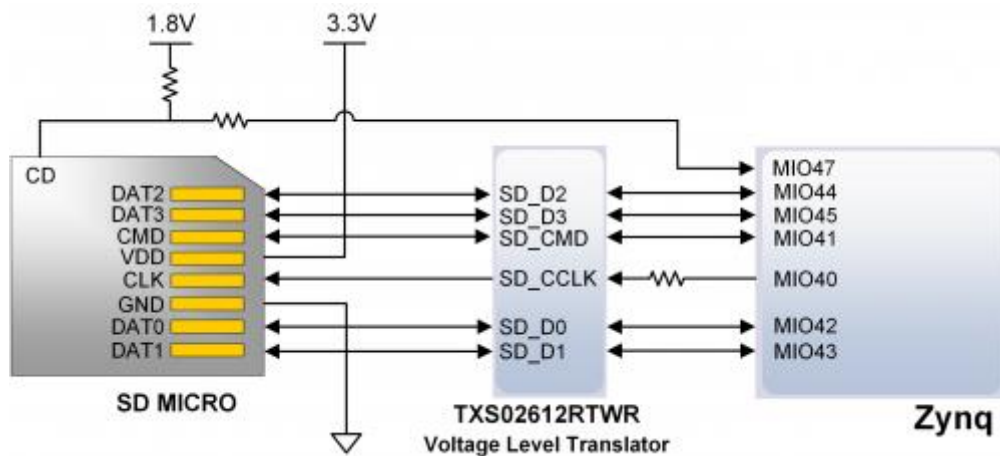


Figure 7.1. microSD slot signals.

Both low speed and high speed cards are supported, the maximum clock frequency being 50 MHz. A Class 4 card or better is recommended.

Refer to section 3.1 for information on how to boot from an SD card. For more information, consult the [Zynq Technical Reference manual](#).

8 USB Host

The PYNQ-Z1 implements one of the two available PS USB OTG interfaces on the Zynq device. A Microchip USB3320 USB 2.0 Transceiver Chip with an 8-bit ULPI interface is used as the PHY. The PHY features a complete HS-USB Physical Front-End supporting speeds of up to 480Mbps. The PHY is connected to MIO Bank 1/501, which is powered at 1.8V. The usb0 peripheral is used on the PS, connected through MIO[28-39]. The USB OTG interface is configured to act as an embedded host. USB OTG and USB device modes are not supported.

The PYNQ-Z1 is technically an “embedded host”, because it does not provide the required 150 μ F of capacitance on VBUS required to qualify as a general purpose host. It is possible to modify the PYNQ-Z1 so that it complies with the general purpose USB host requirements by loading C41 with a 150 μ F capacitor. Only those experienced at soldering small components on PCBs should attempt this rework. Many USB peripheral devices will work just fine without loading C41. Whether the PYNQ-Z1 is configured as an embedded host or a general purpose host, it can provide 500 mA on the 5V VBUS line. Note that loading C41 may cause the PYNQ-Z1 to reset when booting embedded Linux while powered from the USB port, regardless of if any USB device is connected to the host port. This is caused by the in-rush current that C41 causes when the USB host controller is enabled and the VBUS power switch (IC9) is turned on.

Note that if your design uses the USB Host port (embedded or general purpose), then the PYNQ-Z1 should be powered via a battery or wall adapter capable of providing more power (such as the one included in the PYNQ-Z1 accessory kit).

9 Ethernet PHY

The PYNQ-Z1 uses a Realtek RTL8211E-VL PHY to implement a 10/100/1000 Ethernet port for network connection. The PHY connects to MIO Bank 501 (1.8V) and interfaces to the Zynq-7000 APSoC via RGMII for data and MDIO for management. The auxiliary interrupt (INTB) and reset (PHYRSTB) signals connect to MIO pins MIO10 and MIO9, respectively.

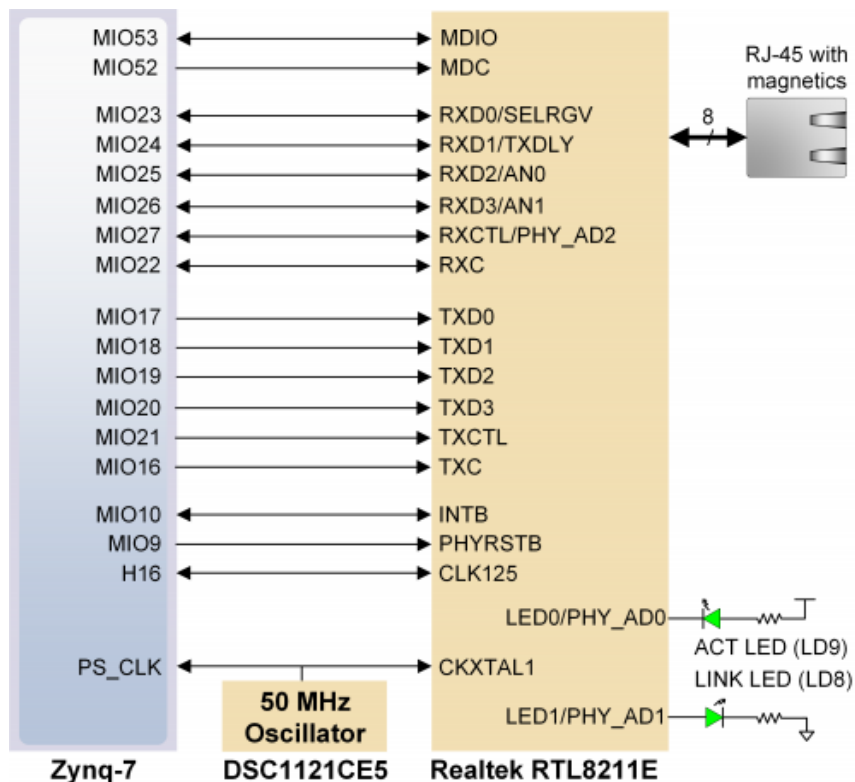


Figure 9.1. Ethernet PHY signals.

After power-up the PHY starts with Auto Negotiation enabled, advertising 10/100/1000 link speeds and full duplex. If there is an Ethernet-capable partner connected, the PHY automatically establishes a link with it, even with the Zynq not configured.

Two status indicator LEDs are on-board near the RJ-45 connector that indicate traffic (LD9) and valid link state (LD8). Table 9.1 shows the default behavior.

Function	Designator	State	Description
LINK	LD8	c	Link 10/100/1000
		Blinking 0.4s ON, 2s OFF	Link, Energy Efficient Ethernet (EEE) mode
ACT	LD9	Blinking	Transmitting or Receiving

Table 9.1. Ethernet status LEDs.

The Zynq incorporates two independent Gigabit Ethernet Controllers. They implement a 10/100/1000 half/full duplex Ethernet MAC. Of these two, GEM 0 can be mapped to the MIO pins where the PHY is connected. Since the MIO bank is powered from 1.8V, the RGMII interface uses 1.8V HSTL Class 1 drivers. For this I/O standard an external reference of 0.9V is provided in bank 501 (PS_MIO_VREF). Mapping out the correct pins and configuring the interface is handled by the PYNQ-Z1 Zynq Presets file, available on the [PYNQ-Z1 Resource Center](#).

Although the default power-up configuration of the PHY might be enough in most applications, the MDIO bus is available for management. The RTL8211E-VL is assigned the 5-bit address 00001 on the MDIO bus. With simple register read and write commands, status information can be read out or configuration changed. The Realtek PHY follows industry-standard register map for basic configuration.

The RGMII specification calls for the receive (RXC) and transmit clock (TXC) to be delayed relative to the data signals (RXD[0:3], RXCTL and TXD[0:3], TXCTL). Xilinx PCB guidelines also require this delay to be added. The RTL8211E-VL is capable of inserting a 2ns delay on both the TXC and RXC so that board traces do not need to be made longer.

The PHY is clocked from the same 50 MHz oscillator that clocks the Zynq PS. The parasitic capacitance of the two loads is low enough to be driven from a single source.

On an Ethernet network each node needs a unique MAC address. To this end, the one-time-programmable (OTP) region of the Quad-SPI flash has been programmed at the factory with a 48-bit globally unique EUI-48/64™ compatible identifier. The OTP address range [0x20;0x25] contains the identifier with the first byte in transmission byte order being at the lowest address. Refer to the [Flash memory datasheet](#) for information on how to access the OTP regions. When using the PYNQ software, this is automatically handled in the boot-loader, and the Linux system is automatically configured to use this unique MAC address.

For more information on using the Gigabit Ethernet MAC, refer to the [Zynq Technical Reference manual](#).

10 HDMI

The PYNQ-Z1 contains two unbuffered HDMI ports: one source port J11 (output), and one sink port J10 (input). Both ports use HDMI type-A receptacles with the data and clock signals terminated and connected directly to the Zynq PL.

Both HDMI and DVI systems use the same TMDS signaling standard, directly supported by Zynq PL's user I/O infrastructure. Also, HDMI sources are backward compatible with DVI sinks, and vice versa. Thus, simple passive adaptors (available at most electronics stores) can be used to drive a DVI monitor or accept a DVI input. The HDMI receptacle only includes digital signals, so only DVI-D mode is possible.

The 19-pin HDMI connectors include three differential data channels, one differential clock channel five GND connections, a one-wire Consumer Electronics Control (CEC) bus, a two-wire Display Data Channel (DDC) bus that is essentially an I2C bus, a Hot Plug Detect (HPD) signal, a 5V signal capable of delivering up to 50mA, and one reserved (RES) pin. All non-power signals are wired to the Zynq PL with the exception of RES.

Pin/Signal	J11 (source)		J10 (sink)	
	Description	FPGA pin	Description	FPGA pin
D[2]_P, D[2]_N	Data output	J18, H18	Data input	N20, P20
D[1]_P, D[1]_N	Data output	K19, J19	Data input	T20, U20
D[0]_P, D[0]_N	Data output	K17, K18	Data input	V20, W20
CLK_P, CLK_N	Clock output	L16, L17	Clock input	N18, P19
CEC	Consumer Electronics Control bidirectional	G15	Consumer Electronics Control bidirectional	H17

Pin/Signal	J11 (source)		J10 (sink)	
	Description	FPGA pin	Description	FPGA pin
SCL, SDA	DDC bidirectional	M17, M18	DDC bidirectional	U14, U15
HPD/HPA	Hot-plug detect input (inverted)	R19	Hot-plug assert output	T19

Table 10.1. HDMI pin description and assignment.

10.1 TMDS Signals

HDMI/DVI is a high-speed digital video stream interface using transition-minimized differential signaling (TMDS). To make proper use of either of the HDMI ports, a standard-compliant transmitter or receiver needs to be implemented in the Zynq PL. The implementation details are outside the scope of this manual. Check out the vivado-library IP Core repository on the [Digilent github](#) for ready-to-use reference IP.

10.2 Auxiliary signals

Whenever a sink is ready and wishes to announce its presence, it connects the 5V0 supply pin to the HPD pin. On the PYNQ-Z1, this is done by driving the Hot Plug Assert signal high. Note this should only be done after a DDC channel slave has been implemented in the Zynq PL and is ready to transmit display data.

The Display Data Channel, or DDC, is a collection of protocols that enable communication between the display (sink) and graphics adapter (source). The DDC2B variant is based on I2C, the bus master being the source and the bus slave the sink. When a source detects high level on the HPD pin, it queries the sink over the DDC bus for video capabilities. It determines whether the sink is DVI or HDMI-capable and what resolutions are supported. Only afterwards will video transmission begin. Refer to VESA E-DDC specifications for more information.

The Consumer Electronics Control, or CEC, is an optional protocol that allows control messages to be passed around on an HDMI chain between different products. A common use case is a TV passing control messages originating from a universal remote to a DVR or satellite receiver. It is a one-wire protocol at 3.3V level connected to a Zynq PL user I/O pin. The wire can be controlled in an open-drain fashion allowing for multiple devices sharing a common CEC wire. Refer to the CEC addendum of HDMI 1.3 or later specifications for more information.

11 Clock Sources

The PYNQ-Z1 provides a 50 MHz clock to the Zynq PS_CLK input, which is used to generate the clocks for each of the PS subsystems. The 50 MHz input allows the processor to operate at a maximum frequency of 650 MHz and the DDR3 memory controller to operate at a maximum of 525 MHz (1050 Mbps). The PYNQ-Z1 Zynq Presets file available on the [PYNQ-Z1 Resource Center](#) can be imported into the Zynq Processing System IP core in a Vivado project to properly configure the Zynq to work with the 50 MHz input clock.

The PS has a dedicated PLL capable of generating up to four reference clocks, each with settable frequencies, that can be used to clock custom logic implemented in the PL. Additionally, the PYNQ-Z1 provides an external 125 MHz reference clock directly to pin H16 of the PL. The external reference clock allows the PL to be used completely independently of the PS, which can be useful for simple applications that do not require the processor.

The PL of the Zynq-Z7020 also includes 4 MMCM's and 4 PLL's that can be used to generate clocks with precise frequencies and phase relationships. Any of the four PS reference clocks or the 125 MHz external reference clock can be used as an input to the MMCMs and PLLs. For a full description of the capabilities of the Zynq PL clocking resources, refer to the "7 Series FPGAs Clocking Resources User Guide" available from Xilinx.

Figure 11.1 outlines the clocking scheme used on the PYNQ-Z1. Note that the reference clock output from the Ethernet PHY is used as the 125 MHz reference clock to the PL, in order to cut the cost of including a dedicated oscillator for this purpose. Keep in mind that CLK125 will be disabled when the Ethernet PHY (IC1) is held in hardware reset by driving the PHYRSTB signal low.

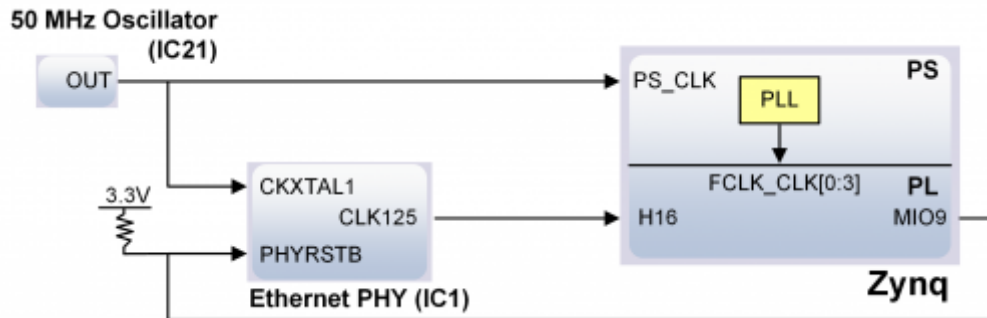


Figure 11.1. PYNQ-Z1 clocking.

12 Basic I/O

The PYNQ-Z1 board includes two tri-color LEDs, 2 switches, 4 push buttons, and 4 individual LEDs as shown in Figure 12.1. The push buttons and slide switches are connected to the Zynq PL via series resistors to prevent damage from inadvertent short circuits (a short circuit could occur if an FPGA pin assigned to a push button or slide switch was inadvertently defined as an output). The four push buttons are “momentary” switches that normally generate a low output when they are at rest, and a high output only when they are pressed. Slide switches generate constant high or low inputs depending on their position.

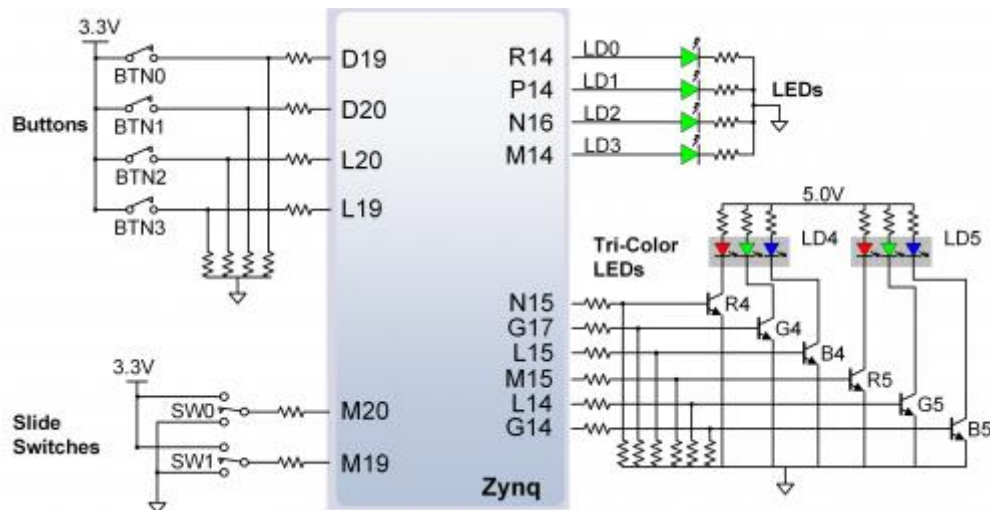


Figure 12.1. PYNQ-Z1 GPIO.

The four individual high-efficiency LEDs are anode-connected to the Zynq PL via 330-ohm resistors, so they will turn on when a logic high voltage is applied to their respective I/O pin. Additional LEDs that are not user-accessible indicate power-on, PL programming status, and USB and Ethernet port status.

12.1 Tri-Color LEDs

The PYNQ-Z1 board contains two tri-color LEDs. Each tri-color LED has three input signals that drive the cathodes of three smaller internal LEDs: one red, one blue, and one green. Driving the signal corresponding to one of these colors high will illuminate the internal LED. The input signals are driven by the Zynq PL through a transistor, which inverts the signals. Therefore, to light up the tri-color LED, the corresponding signals need to be driven high. The tri-color LED will emit a color dependent on the combination of internal LEDs that are currently being illuminated. For example, if the red and blue signals are driven high and green is driven low, the tri-color LED will emit a purple color.

Digilent strongly recommends the use of pulse-width modulation (PWM) when driving the tri-color LEDs. Driving any of the inputs to a steady logic '1' will result in the LED being illuminated at an uncomfortably bright level. You can avoid this by ensuring that none of the tri-color signals are driven with more than a 50% duty cycle. Using PWM also greatly expands the potential color palette of the tri-color led. Individually adjusting the duty cycle of each color between 50% and 0% causes the different colors to be illuminated at different intensities, allowing virtually any color to be displayed.

13 Microphone

The PYNQ-Z1 includes an omnidirectional MEMS microphone. The microphone uses a Knowles SPK0833LM4H-B chip which has a high signal to noise ratio (SNR) of 63dBa and high sensitivity of -26 dBFS. The digitized audio is output in the pulse density modulated (PDM) format. The component architecture is shown in Figure 13.1.



Figure 13.1. Microphone Pinout.

13.1 Pulse Density Modulation (PDM)

PDM data connections are becoming more and more popular in portable audio applications, such as cellphones and tablets. With PDM, two channels can be transmitted with only two wires. The frequency of a PDM signal usually falls in the range of 1 MHz to 3 MHz. In a PDM bitstream, a 1 corresponds to a positive pulse and a 0 corresponds to a negative pulse. A run consisting of all '1's would correspond to the maximum positive value and a run of '0's would correspond to the minimum amplitude value. Figure 13.1.1 shows how a sine wave is represented in PDM signal.

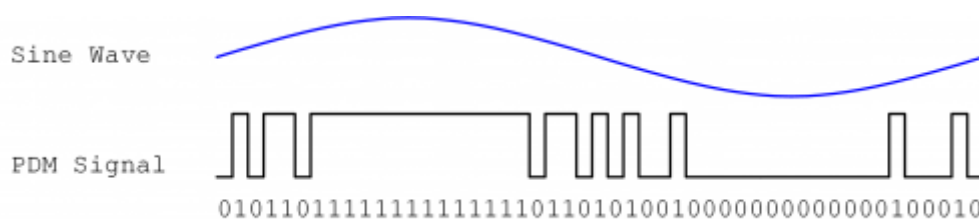


Figure 13.1.1. PDM Sine Wave.

A PDM signal is generated from an analog signal through a process called delta-sigma modulation. A simple idealized circuit of delta-sigma modulator is shown in Figure 13.1.2.

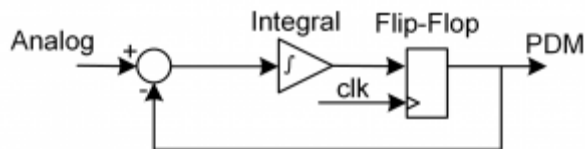


Figure 13.1.2. PDM Delta-Sigma Modulator.

Sum	Integrator Out	Flip-flop Output
0.4-0=0.4	0+0.4=0.4	0
0.4-0=0.4	0.4+0.4=0.8	1
0.4-1=-0.6	0.8-0.6=0.2	0
0.4-0=0.4	0.2+0.4=0.6	1
0.4-1=-0.6	0.6-0.6=0	0
0.4-0=0.4	0+0.4=0.4	0
0.4-0=0.4	0.4+0.4=0.8	1
0.4-1=-0.6	0.8-0.6=0.2	0

Table 13.1.1. Sigma Delta Modulator with a 0.4V_{dd} input.

To keep things simple, assume that the analog input and digital output have the same voltage range 0~V_{dd}. The input of the flip-flop acts like a comparator (any signal above V_{dd}/2 is considered as ‘1’ and any input below V_{dd}/2 is considered ‘0’). The input of the integral circuit is the difference of the input analog signal and the PDM signal of the previous clock cycle. The integral circuit then integrates both of these inputs, and the output of the integral circuit is sampled by a D-Flip-flop. Table 13.1.1 shows the function of the delta-sigma modulator with an input of 0.4V_{dd}.

Note that the average of the flip-flop output equals the value of the input analog signal. So in order to get the value of analog input, all that is needed is a counter that counts the ‘1’s for a certain period of time.

13.2 Microphone Digital Interface Timing

The clock input of the microphone can range from 1 MHz to 3.3 MHz based on the sampling rate and data precision requirement of the applications. The L/R Select signal must be set to a valid level, depending on which edge of the clock the data bit will be read. A low level on L/RSEL makes data available on the rising edge of the clock, while a high level corresponds to the falling edge of the clock, as shown in Fig. 13.2.1. Note that on the PYNQ-Z1, the L/RSEL signal is permanently tied low, so data is always made available on the rising edge (as seen with the DATA 1 signal in Fig. 13.2.1).

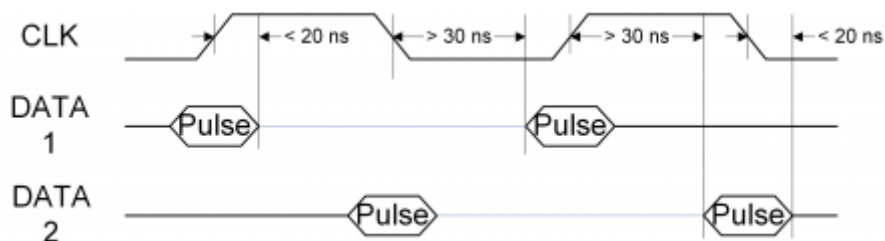


Figure 13.2.1. Microphone Signal Timing.

The typical value of the clock frequency is 2.4 MHz. Assuming that the application requires 7-bit precision and 24 KHz, there can be two counters that count 128 samples at 12 KHz, as shown in Fig. 13.2.2.

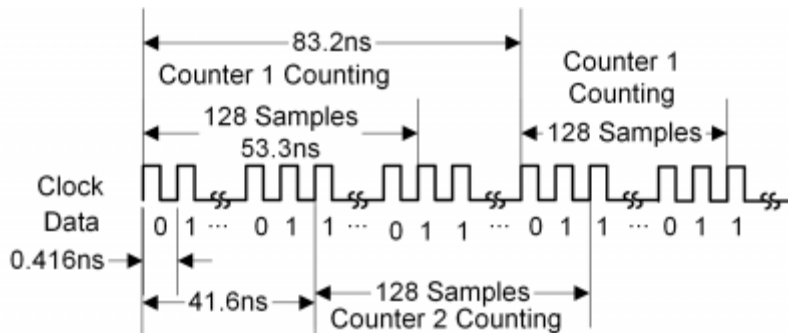


Figure 13.2.2. Example PDM Circuit.

14 Mono Audio Output

The on-board audio jack (J13) is driven by a Sallen-Key Butterworth Low-pass 4th Order Filter that provides mono audio output. The circuit of the low-pass filter is shown in Fig. 14.1. The input of the filter (AUD_PWM) is connected to the Zynq PL pin R18. A digital input will typically be a pulse-width modulated (PWM) or pulse density modulated (PDM) open-drain signal produced by the FPGA. The signal needs to be driven low for logic '0' and left in high-impedance for logic '1'. An on-board pull-up resistor to a clean analog 3.3V rail will establish the proper voltage for logic '1'. The low-pass filter on the input will act as a reconstruction filter to convert the pulse-width modulated digital signal into an analog voltage on the audio jack output.

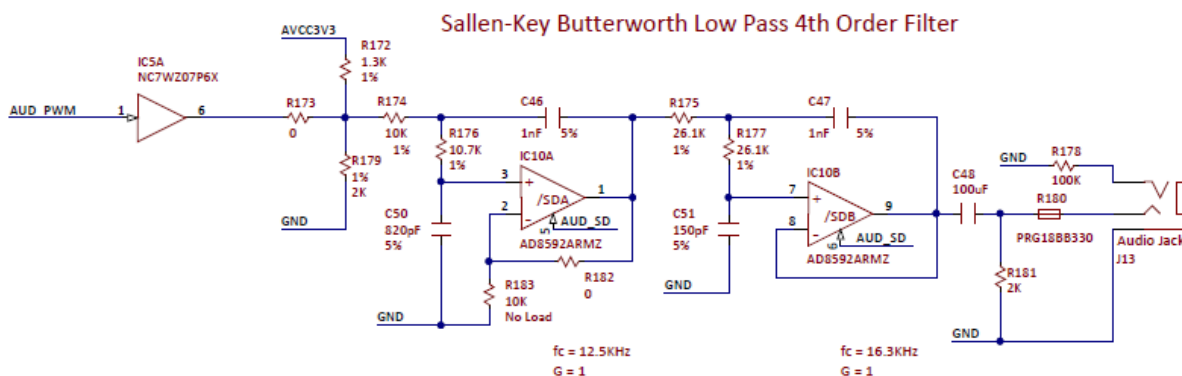


Figure 14.1. Audio Output Circuit.

The Audio shut-down signal (AUD_SD) is used to mute the audio output. It is connected to Zynq PL pin T17. To use the audio output, this signal must be driven to logic high.

The frequency response of SK Butterworth Low-Pass Filter is shown in Fig. 14.2. The AC analysis of the circuit is done using NI Multisim 12.0.

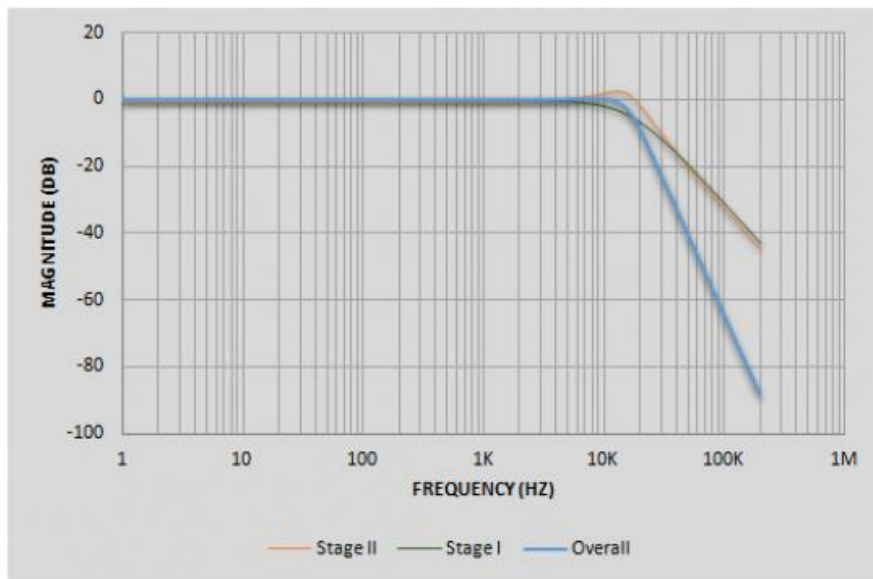


Figure 14.2. Audio Output Frequency Response.

14.1 Pulse-Width Modulation

A pulse-width modulated (PWM) signal is a chain of pulses at some fixed frequency, with each pulse potentially having a different width. This digital signal can be passed through a simple low-pass filter that integrates the digital waveform to produce an analog voltage proportional to the average pulse-width over some interval (the interval is determined by the 3dB cut-off frequency of the low-pass filter and the pulse frequency). For example, if the pulses are high for an average of 10% of the available pulse period, then an integrator will produce an analog value that is 10% of the V_{dd} voltage. Figure 14.1.1 shows a waveform represented as a PWM signal.

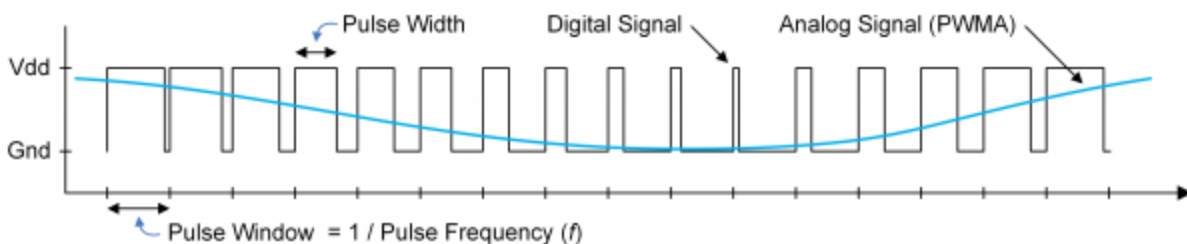


Figure 14.1.1. PWM Waveform.

The PWM signal must be integrated to define an analog voltage. The low-pass filter 3dB frequency should be an order of magnitude lower than the PWM frequency, so that signal energy at the PWM frequency is filtered from the signal. For example, if an audio signal must contain up to 5 KHz of frequency information, then the PWM frequency should be at least 50 KHz (and preferably even higher). In general, in terms of analog signal fidelity, the higher the PWM frequency, the better. Figure 14.1.2 shows a representation of a PWM integrator producing an output voltage by integrating the pulse train. Note the steady-state filter output signal amplitude ratio to V_{dd} is the same as the pulse-width duty cycle (duty cycle is defined as pulse-high time divided by pulse-window time).

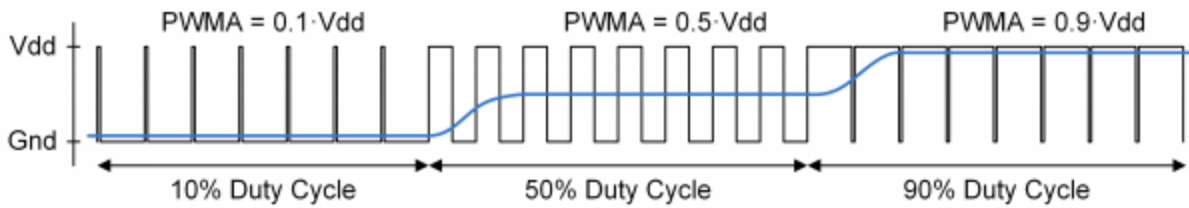


Figure 14.1.2. PWM Output Voltage.

15 Reset Sources

15.1 Power-on Reset

The Zynq PS supports external power-on reset signals. The power-on reset is the master reset of the entire chip. This signal resets every register in the device capable of being reset. The PYNQ-Z1 drives this signal from the PGOOD signal of the TPS65400 power regulator in order to hold the system in reset until all power supplies are valid.

15.2 Program Push Button Switch

A PROG push switch, labeled PROG, toggles Zynq PROG_B. This resets the PL and causes DONE to be de-asserted. The PL will remain unconfigured until it is reprogrammed by the processor or via JTAG.

15.3 Processor Subsystem Reset

The external system reset, labeled SRST, resets the Zynq device without disturbing the debug environment. For example, the previous break points set by the user remain valid after system reset. Due to security concerns, system reset erases all memory content within the PS, including the OCM. The PL is also cleared during a system reset. System reset does not cause the boot mode strapping pins to be re-sampled.

The SRST button also causes the CK_RST signal to toggle in order to trigger a reset on any attached shields.

16 Pmod Ports

Pmod ports are 2×6, right-angle, 100-mil spaced female connectors that mate with standard 2×6 pin headers. Each 12-pin Pmod port provides two 3.3V VCC signals (pins 6 and 12), two Ground signals (pins 5 and 11), and eight logic signals, as shown in Figure 16.1. The VCC and Ground pins can deliver up to 1A of current, but care must be taken not to exceed any of the power budgets of the onboard regulators or the external power supply (see the 3.3V rail current limits listed in the “Power Supplies” section).

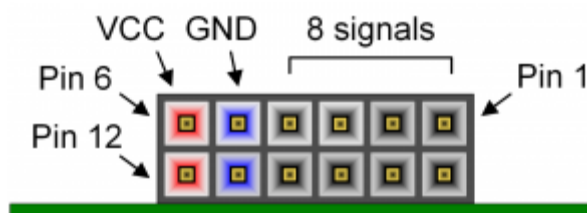


Figure 16.1. Pmod Port Diagram.

Digilent produces a large collection of Pmod accessory boards that can attach to the Pmod expansion connectors to add ready-made functions like A/D's, D/A's, motor drivers, sensors, and other functions.

Each Pmod port found on Digilent FPGA boards falls into one of four categories: standard, MIO connected, XADC, or high-speed. The PYNQ-Z1 has two Pmod ports, both of which are the high-speed type. The following section describes the high-speed type of Pmod port.

16.1 High-Speed Pmods

The High-speed Pmods have their data signals routed as impedance matched differential pairs for maximum switching speeds. They have pads for loading resistors for added protection, but the PYNQ-Z1 ships with these loaded as 0-Ohm shunts. With the series resistors shunted, these Pmods offer no protection against short circuits, but allow for much faster switching speeds. The signals are paired to the adjacent signals in the same row: pins 1 and 2, pins 3 and 4, pins 7 and 8, and pins 9 and 10.

Traces are routed 100 ohm (+/- 10%) differential.

If pins on this port are used as single-ended signals, coupled pairs may exhibit crosstalk. In applications where this is a concern, one of the signals should be grounded (drive it low from the FPGA) and use its pair for the signal-ended signal.

Since the High-Speed Pmods have 0-ohm shunts instead of protection resistors, the operator must take precaution to ensure that they do not cause any shorts.

17 Arduino/chipKIT Shield Connector

The PYNQ-Z1 can be connected to standard Arduino and chipKIT shields to add extended functionality. Special care was taken while designing the PYNQ-Z1 to make sure it is compatible with the majority of Arduino and chipKIT shields on the market. The shield connector has 49 pins connected to the Zynq PL for general purpose Digital I/O. Due to the flexibility of FPGAs, it is possible to use these pins for just about anything including digital read/write, SPI connections, UART connections, I2C connections, and PWM. Six of these pins (labeled AN0-AN5) can also be used as single-ended analog inputs with an input range of 0V-3.3V, and another six (labeled AN6-11) can be used as differential analog inputs.

Note: The PYNQ-Z1 is not compatible with shields that output 5V digital or analog signals. Driving pins on the PYNQ-Z1 shield connector above 5V may cause damage to the Zynq.

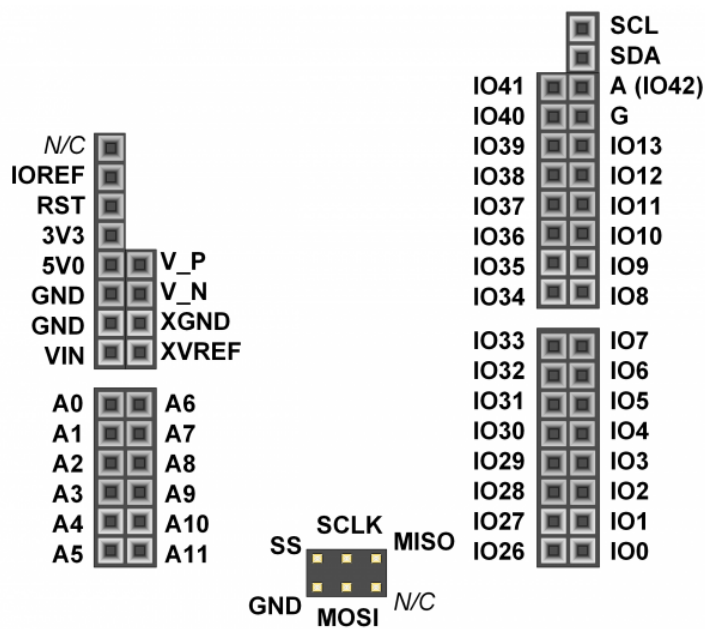


Figure 17.1. Shield Pin Diagram.

Pin Name	Shield Function	PYNQ-Z1 Connection
IO0-IO13, IO26-IO41, A (IO42)	General purpose I/O pins	See Section titled “Shield Digital I/O”
SCL	I2C Clock	See Section titled “Shield Digital I/O”
SDA	I2C Data	See Section titled “Shield Digital I/O”
SCLK	SPI Clock	See Section titled “Shield Digital I/O”
MOSI	SPI Data out	See Section titled “Shield Digital I/O”
MISO	SPI Data in	See Section titled “Shield Digital I/O”
SS	SPI Slave Select	See Section titled “Shield Digital I/O”
A0-A5	Single-Ended Analog Input	See Section titled “Shield Analog I/O”
A6-A11	Differential Analog Input	See Section titled “Shield Analog I/O”
V_P, V_N	Dedicated Differential Analog Input	See Section titled “Shield Analog I/O”
XGND	XADC Analog Ground	Connected to net used to drive the XADC ground reference on the Zynq (VREFN)

Pin Name	Shield Function	PYNQ-Z1 Connection
XVREF	XADC Analog Voltage Reference	Connected to 1.25 V, 25mA rail used to drive the XADC voltage reference on the Zynq (VREFP)
N/C	Not Connected	Not Connected
IOREF	Digital I/O Voltage reference	Connected to the PYNQ-Z1 3.3V Power Rail (See the “Power Supplies” section)
RST	Reset to Shield	Connected to the red “SRST” button and MIO pin 12 of the Zynq. When JP1 is shorted, it is also connected to the DTR signal of the FTDI USB-UART bridge.
3V3	3.3V Power Rail	Connected to the PYNQ-Z1 3.3V Power Rail (See the “Power Supplies” section)
5V0	5.0V Power Rail	Connected to the PYNQ-Z1 5.0V Power Rail (See the “Power Supplies” section)
GND, G	Ground	Connected to the Ground plane of PYNQ-Z1
VIN	Power Input	Connected in parallel with the external power supply connector (J18).

Table 17.1. Shield Pin Descriptions.

17.1 Shield Digital I/O

The pins connected directly to the Zynq PL can be used as general purpose inputs or outputs. These pins include the I2C, SPI, and general purpose I/O pins. There are 200 Ohm series resistors between the FPGA and the digital I/O pins to help provide protection against accidental short circuits (with the exception of the AN5-AN0 signals, which have no series resistors, and the AN6-AN12 signals, which have 100 Ohm series resistors). The absolute maximum and recommended operating voltages for these pins are outlined in the table below.

	Absolute Minimum Voltage	Recommended Minimum Operating Voltage	Recommended Maximum Operating Voltage	Absolute Maximum Voltage
Powered	-0.4 V	-0.2 V	3.4 V	3.75 V
Unpowered	-0.4 V	N/A	N/A	0.55 V

Table 17.1.1. Shield Digital Voltages.

For more information on the electrical characteristics of the pins connected to the Zynq PL, please see the [Zynq-7000 datasheet](#) from Xilinx.

17.2 Shield Analog I/O

The pins labeled A0-A11 and V_P/V_N are used as analog inputs to the XADC module of the Zynq. The Zynq expects that the inputs range from 0-1 V. On the pins labeled A0-A5 we use an external circuit to scale down the input voltage from 3.3V. This circuit is shown in Fig. 17.2.1. This circuit allows the XADC module to accurately measure any voltage between 0V and 3.3V (relative to the PYNQ-Z1's GND) that is applied to any of these pins. If you wish to use the pins labeled A0-A5 as Digital inputs or outputs, they are also connected directly to the Zynq PL before the resistor divider circuit (also shown in Fig. 17.2.1).

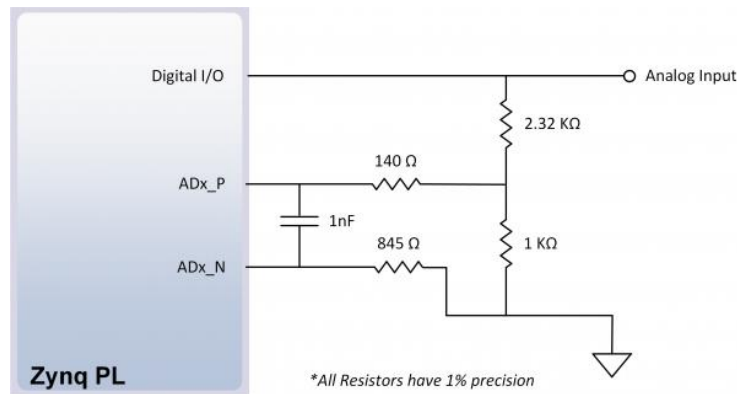


Figure 17.2.1. Single-Ended Analog Inputs.

The pins labeled A6-A11 are connected directly to 3 pairs of analog capable pins on the Zynq PL via an anti-aliasing filter. This circuit is shown in Fig. 17.2.2. These pairs of pins can be used as differential analog inputs with a voltage difference between 0-1V. The even numbers are connected to the positive pins of the pair and the odd numbers are connected to the negative pins (so A6 and A7 form an analog input pair with A6 being positive and A7 being negative). Note that though the pads for the capacitor are present, they are not loaded for these pins. Since the analog capable pins of the FPGA can also be used like normal digital FPGA pins, it is also possible to use these pins for Digital I/O.

The pins labeled V_P and V_N are connected to the VP_0 and VN_0 dedicated analog inputs of the FPGA. This pair of pins can also be used as a differential analog input with voltage between 0-1V, but they cannot be used as Digital I/O. The capacitor in the circuit shown in Fig. 17.2.2 for this pair of pins is loaded on the PYNQ-Z1.

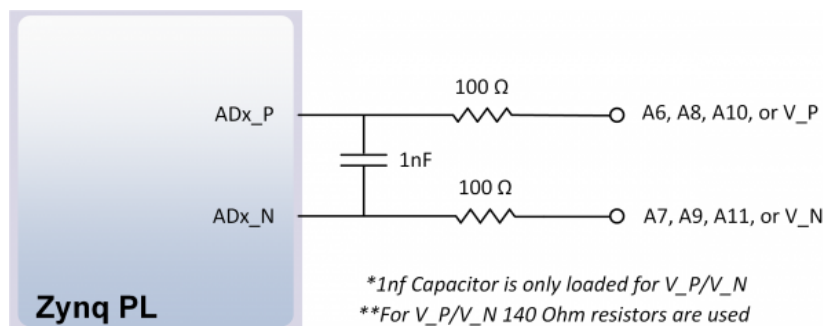


Figure 17.2.2. Differential Analog Inputs.

The XADC core within the Zynq is a dual channel 12-bit analog-to-digital converter capable of operating at 1 MSPS. Either channel can be driven by any of the analog inputs connected to the shield pins. The XADC core is controlled and accessed from a user design via the Dynamic Reconfiguration Port (DRP). The DRP also provides access to voltage monitors that are present on each of the FPGA's power rails, and a temperature sensor that is internal to