ball.sv

```
Inputs:
    Reset
    frame_clk
    [7:0] keycode

Outputs:
    [9:0] BallX, BallY, BallS
```

Description: This module simulates the movement of a ball on a 2D grid. The ball has a predefined size (`BallS`) and can move within a confined space defined by maximum and minimum X and Y positions (`Ball_X_Max`, `Ball_X_Min`, `Ball_Y_Max`, `Ball_Y_Min`). Its motion is determined by the `keycode` (W/A/S/D) input which corresponds to different directional controls.

- The `Reset` signal initializes the ball to its center position (`Ball_X_Center`, `Ball_Y_Center`).
- The ball moves with a specific step size, defined by `Ball_X_Step` and `Ball_Y_Step`, in the X or Y direction based on the input `keycode`.
- Boundary checks ensure the ball doesn't cross the predefined limits and modifies its movement accordingly.
- All motion updates are done on the rising edge of the `frame_clk` signal.

Purpose: To simulate the motion of a ball on a grid with user controls provided by the `keycode`. The output is then used by color mapper to generate the pixels of the ball and background

Color_Mapper.sv

```
Inputs:
    [9:0] BallX, BallY, DrawX, DrawY, Ball_size

Outputs:
    [3:0] Red, Green, Blue
```

Description: The `color_mapper` module maps pixel positions to specific color (yellowish) values if pixel is inside the ball, otherwise to a gradient color depending where the pixel is.

Purpose: Transform the coordinate of the ball, to the pixel color used for VGA signal output (later transformed via VGA-HDMI IP).

vga_controller.sv

```
Inputs:
    pixel_clk        // 50 MHz clock
    reset            // Reset signal

Outputs:
    hs               // Horizontal sync pulse. Active low
    vs               // Vertical sync pulse. Active low
    active_nblank    // High = active, low = blanking interval
    sync             // Composite Sync signal. Active low (Not used in this lab
but required for DE2 board's video DAC)
    [9:0] drawX      // Horizontal coordinate
    [9:0] drawY      // Vertical coordinate
```

Description: The `vga_controller` module generates synchronization signals required for a VGA display. It operates on an 800x525 pixel screen but confines active display to a 640x480 pixel region. The module generates horizontal and vertical sync pulses, ensuring that pixels are drawn at correct coordinates on the screen.

Purpose: The module is crucial for interfacing with VGA displays. It ensures synchronization, dictating when and where pixels should be drawn, enabling the creation of stable images on VGA screens. These control pulses are later given to the VGA-HDMI IP

VGA-HDMI IP

```
Inputs:
    pix_clk, pix_clkx5, pix_clk_locked
    reset_ah, red, green, blue,
    hsync, vsync, vde,

    aux0_din, aux1_din, aux2_din, ade

Outputs:
    [2:0] TMDS_CLK_P
    [2:0] TMDS_CLK_N
    TMDS_DATA_P
    TMDS_DATA_N
```

Description: The `hdmi_tx_0` module acts as a VGA to HDMI converter. It takes in VGA signals such as red, green, blue color channels along with sync signals (hsync and vsync) and converts them into differential HDMI outputs. The pixel clock and its multiplied version (5x) drive the conversion process. A separate locked signal indicates the status of the pixel clock. The module also provides auxiliary data inputs, which are unused in this context and are set to default values.

Purpose: This module serves as a bridge between VGA interfaces and HDMI outputs, facilitating the conversion of VGA signals into HDMI format.

mb_usb_hdmi_top.sv

```
Inputs:
    Clk
    reset_rtl_0
    gpio_usb_int_tri_i
    usb_spi_miso
    uart_rtl_0_rxd

Outputs:
    gpio_usb_rst_tri_o
    usb_spi_mosi
    usb_spi_sclk
    usb_spi_ss
    uart_rtl_0_txd
    hdmi_tmds_clk_n
    hdmi_tmds_clk_p
    [2:0]hdmi_tmds_data_n
    [2:0]hdmi_tmds_data_p
    [7:0] hex_segA
    [3:0] hex_gridA
    [7:0] hex_segB
    [3:0] hex_gridB
```

Description: This top-level module, `mb_usb_hdmi_top`, integrates various sub-modules like USB, UART, and HDMI functionalities. It also interacts with HEX displays. The USB-related signals are primarily handled by the `mb_block` instance (which has MicroBlaze processor and necessary peripherals), which interfaces with the external USB signals and produces keycodes. The HDMI-related operations are managed by the `vga_controller` and `hdmi_tx_0` instances, which convert VGA signals to HDMI. The `ball` and `color_mapper` modules work together to produce color outputs based on the ball's position. The `HexDriver` instances are responsible for driving the HEX displays.

Purpose: This module is the top-level construct. It integrates various functionalities, including USB interactions, VGA to HDMI conversion, ball position-based color mapping, and HEX display drivers.

HexDriver.sv

```
Inputs:
    clk, reset,
    [3:0] in[4],
Outputs:
    [7:0] hex_seg,
    [3:0] hex_grid
```

Description: Given 4 x **4-bit** data `[3:0] in[4]`, it will generate correct `hex_seg` and `hex_grid` which will be used to display data on the FPGA board.

Purpose: Display the 8-bit logic processor's register on the FPGA's Segment LED display.

Clock Wizard IP

```
Inputs:
    Clk         - Primary input clock source.
    reset_ah    - Active high reset signal.

Outputs:
    clk_25MHz   - Output clock signal with frequency 25MHz.
    clk_125MHz  - Output clock signal with frequency 125MHz.
    locked      - Indicates whether the clock outputs are stable and running at
their intended frequencies.
```

Description: The `clk_wiz_0` module is a clock generator/wizard, responsible for producing different clock frequencies derived from a primary input clock, `Clk`. It provides two clock outputs: `clk_25MHz` and `clk_125MHz`. The module can be reset using the `reset_ah` signal. When the output clocks are stable and functioning at their expected frequencies, the `locked` signal is asserted.

Purpose: This module is utilized to generate different clock frequencies required for various parts of the design. It ensures synchronized operations across multiple components by providing clocks derived from a single source.

Clocking Wizard (Vivado IP)

Description: The Clocking Wizard in Vivado is a customizable IP core designed to generate multiple clock frequencies derived from a primary reference clock. It utilizes Phase-Locked Loops (PLLs) and/or Digital Clock Managers (DCMs) depending on the FPGA family and user configuration. The Clocking Wizard allows users to specify the desired output frequencies, phase shifts, and duty cycles. Furthermore, it provides features to optimize jitter performance and introduces protection mechanisms against clock glitches.

Purpose: The primary goal of the Clocking Wizard is to generate various clock frequencies required for a design from a single reference clock. This aids in maintaining synchronization across different parts of an FPGA design and provides flexibility in clock configuration to meet specific design requirements. It is an essential IP in Vivado for designs that demand multiple clock domains or specific clock characteristics.

Processor System Reset IP

Description: The Processor System Reset IP core in Vivado for the Spartan 6 FPGA provides a reliable reset mechanism for processor-based systems in Xilinx FPGAs. It typically manages various reset requirements, including managing the release of the system's reset in the correct sequence. The IP is capable of handling both the assertion and de-assertion of the processor reset signal based on specific conditions, ensuring safe startup and shutdown sequences. Furthermore, it might handle peripherals and external components' reset requirements, ensuring that the processor and its associated elements start operation in a synchronized manner.

Purpose: The main objective of the Processor System Reset IP is to streamline and manage reset sequences within processor-based designs. This becomes especially crucial in complex designs where various components might have interdependencies or require a specific startup sequence. By utilizing this IP, designers can ensure a safe, consistent, and synchronized reset sequence,

enhancing system reliability and robustness.

AXI Interconnect IP

Description: The AXI Interconnect IP in Vivado facilitates the connection between one or multiple AXI memory-mapped Master devices and one or multiple memory-mapped Slave devices. It acts as a communication bridge and can handle AXI4, AXI3, and AXI4-Lite interfaces. The interconnect is highly configurable and supports features such as data width conversion, clock conversion, and different arbitration schemes.

Purpose: The primary objective of the AXI Interconnect IP is to enable communication between different AXI interfaces, especially when multiple Masters need to interact with multiple Slaves. It efficiently manages data traffic, ensuring that data transactions adhere to the AXI protocol standards.

AXI Quad SPI IP

Description: The AXI Quad SPI IP in Vivado is a soft Xilinx IP core that provides Quad Serial Peripheral Interface (SPI) communication, compliant with the AXI4-Lite interface. It supports standard SPI, Dual SPI, and Quad SPI modes. The Dual/Quad SPI is the enhancement to the Standard SPI protocol that delivers a simple method for a master and a selected slave to exchange data.

Purpose: The AXI Quad SPI IP is designed to facilitate SPI communication in FPGA designs, leveraging the AXI4-Lite interface. It's particularly useful for systems that require high-speed data transfer, extended data widths, or communication with multiple SPI devices. This IP abstracts the SPI communication complexities and offers a straightforward integration with AXI-based designs in Xilinx FPGAs.

AXI Interrupt Controller IP

Description: The AXI Interrupt Controller IP in Vivado is designed to manage multiple interrupt sources in a system that utilizes the AXI protocol. This IP core centralizes multiple interrupt inputs and provides a single interrupt output to the processor or another higher-level interrupt controller. It supports prioritization of interrupt sources, enabling more critical interrupts to be processed before others. The controller can be programmed to handle interrupts in specific ways, such as masking certain interrupts or defining the edge on which an interrupt is triggered.

Purpose: The primary role of the AXI Interrupt Controller IP is to consolidate and manage interrupts in AXI-based systems efficiently. By centralizing interrupt management, it simplifies the design and ensures that the processor or main controller can handle interrupts in a structured and prioritized manner. This results in improved system responsiveness and reliability, especially in complex systems with numerous interrupt-generating sources.

AXI Timer IP

Description: The AXI Timer IP in Vivado provides a general-purpose timer solution for AXI-based systems. It supports both capture and generation modes, allowing users to measure time intervals or generate precise time delays. In capture mode, the AXI Timer captures and stores the value of the timer counter when an external input is asserted. In generation mode, the AXI Timer generates an output signal when the timer counter matches a specified value. The IP includes features such as up/down counting and auto-reload capabilities.

Purpose: The AXI Timer IP is used in AXI-based systems to provide accurate timing mechanisms. Whether measuring time intervals with the capture mode or generating precise time delays using generation mode, the AXI Timer is essential for tasks requiring precise timing. Common use cases include waveform generation, frequency measurement, and event timing.

AXI GPIO IP

Description: The AXI GPIO IP in Vivado provides a general-purpose I/O interface to the AXI interface. It allows software running on a processor to control and monitor the GPIO pins. The core can be configured in Vivado to have single or dual channels, and each channel can have any number of pins. In addition, the core can be set up to generate an interrupt when any of its pins change state. When tri-state control is enabled, the `gpio_t` and `gpio2_t` signals can be used to set the GPIO pins as inputs or outputs. The data to be output to the pins is written to the `gpio_o` and `gpio2_o` signals, and data from the pins can be read from the `gpio_i` and `gpio2_i` signals.

Purpose: The AXI GPIO IP is designed to give software control over general-purpose I/O pins through the AXI interface. This is particularly useful for designs that include a processor, as it allows software to interact with the outside world, controlling outputs and reading inputs from the GPIO pins. The IP can be used for a wide variety of tasks, such as reading the state of buttons, driving LEDs, interfacing with custom hardware, and more.

AXI Uartlite IP (Vivado)

Description: The AXI Uartlite IP core provided in Vivado is a compact and efficient UART (Universal Asynchronous Receiver Transmitter) controller, which interfaces with the AXI4-Lite protocol. It facilitates serial communication using standard UART protocols. The IP provides fundamental UART operations, enabling data transmission and reception without the need for a full-fledged UART controller. It also supports configurable data and stop bits.

Purpose: The AXI Uartlite IP is designed for applications that require low-power, area-optimized UART solutions for chip-to-chip communication, debugging, or simple communication tasks. As it follows the AXI4-Lite protocol, it's straightforward to integrate with other AXI-interfaced IP cores in the Vivado environment.