To build a solid understanding of SoC fundamentals and practice functional modelling of the BabySoC using simulation tools (Icarus Verilog & GTKWave).

Part 1 – Theory (Conceptual Understanding) Go through the Fundamentals of SoC Design   
  
notes: Theory Reference:- https://github.com/hemanthkumardm/SFAL-VSD-SoCJourney/tree/main/11.%20Fundamentals%20of%20SoC%20Design

Focus on:

o What is a System-on-Chip (SoC)?

o Components of a typical SoC (CPU, memory, peripherals, interconnect).

o Why BabySoC is a simplified model for learning SoC concepts.

o The role of functional modelling before RTL and physical design stages.

Deliverable: A 1–2 page write-up on GitHub summarizing your understanding of SoC design fundamentals and how BabySoC fits into this learning journey.

1. **Open source System-On-Chip design based on RVMYTH based processor core. The said SoC houses:**
   1. **Phase Lock Loop: - [Clock Generation]**
      1. In **VLSI (Very Large Scale Integration)**, a **PLL (Phase-Locked Loop)** is a closed-loop control system that generates a stable, high-frequency clock signal that is synchronized with a reference clock.
      2. It’s widely used in chip design for clock generation, synchronization, and frequency multiplication.
      3. **Clock Multiplication/Division**
         1. Converts a low-frequency reference clock into a higher-frequency clock (or vice versa).  
             Example: A 50 MHz reference can be multiplied to 500 MHz system clock.
      4. **Clock Jitter Reduction**
         1. Cleans up noisy input clocks and generates a stable clock.
      5. **Clock Recovery**
         1. Used in serial communication (like USB, PCIe, Ethernet) to extract clock information from data.
      6. **Synchronization**
         1. Keeps multiple clock domains in sync.
   2. **10-bit Digital-to-Analog Converter (DAC): - [Interface to outside world]**
      1. in SoC (System-on-Chip) design, DAC (Digital-to-Analog Converter) and ADC (Analog-to-Digital Converter) play a crucial role in bridging the digital and analog worlds.
      2. **Resolution**
         1. With 10 bits, there are 210=10242^{10} = 1024210=1024 possible digital codes.
         2. Each step corresponds to **1 LSB (Least Significant Bit)** change.
      3. **Input**
         1. 10-bit binary input: D9:09:09:0.
         2. Ranges from 0000000000 (0 decimal) to 1111111111 (1023 decimal).
      4. **Output**
         1. Analog output is proportional to digital input:
            1. Vout=D210−1×VrefV\_{out} = \frac{D}{2^{10}-1} \times V\_{ref}Vout​=210−1D​×Vref​  
                where **D** = decimal equivalent of 10-bit input.
      5. **Types of DAC Architectures** (common in VLSI):
         1. **Binary-weighted resistor DAC** (simple but impractical for >8 bits due to resistor size variations).
         2. **R-2R Ladder DAC** (commonly used, requires only two resistor values).
         3. **Current-steering DAC** (fast, used in high-speed applications).
         4. **Sigma-Delta DAC** (high resolution, used in audio).
      6. Why are ADC and DAC needed?
         1. **ADC (Analog → Digital)** Converts **real-world analog signals** (like sound, temperature, voltage, sensor outputs) into **digital values** that the SoC (processor, DSP, or logic) can process.
            1. Examples:

Microphone input → digitized audio for processing.

Temperature sensor → digital reading for monitoring.

Camera sensor → pixel voltages → digital image data.

* + - * 1. **Use in SoC**:

Sensor interfaces (IoT devices, wearables, automotive SoCs).

Wireless communication (receiving RF signals → digitized for baseband processing).

Data acquisition in medical/industrial SoCs.

* + - 1. **DAC (Digital → Analog)** Converts **digital data** inside the SoC into **analog signals** to interact with the real world.
         1. Examples:

Audio playback (digital music → analog speaker/headphone signal).

Video generation (digital pixel data → analog voltage for display driver in some systems).

Control voltages for actuators, motors, or RF circuits.

* + - * 1. **Use in SoC**:

Audio output (headphones, speakers).

Communication transmitters (digital baseband → analog RF signal).

Motor drivers, actuators, control systems.

* + - * 1. dsad
  1. This “Sky130-technology-based” SoC aims to provide a highly documented, educational platform for learning and experimentation in digital-analog interfacing.

1. **A SoC is a mini computer.**
   1. Built on a single chip.
   2. SoC combines everything into one small package.
   3. Useful where space, power and efficiency are important.
   4. Examples:
      1. Smartphones
      2. Smartwatches
      3. Tablets
      4. Cars
      5. Home appliances?
2. **Key Parts of SoC**
   1. **CPU - Central Processing Unit - Brain of the SoC**.  
      1. Manages:
         1. Booting
         2. Configuration
         3. Blocks initialization
         4. Data transfer - usually handled via. DMA.
         5. Handles interrupts - based on ISR - [Interrupt Service Routine]
         6. Does calculations, data processing.
   2. **Memory:**
      1. RAM - Random Access Memory - Temporary Storage.
         1. Two types
            1. SRAM - Static RAM

Stores data using flip-flops - bistable latches.

More area - 6 transistors - faster access.

Used in cache memory inside.

Processors.

Register files.

FIFO.

L1 - one with each core.

L2 - one for the whole processor Sub-system.

L3.

On chip memory.

* + - * 1. DRAM - Dynamic RAM

Slower than SRAM but Area is less.

Stores data using a capacitor + transistor per bit.

Used in main memory.

DDR - double data rate

Sampling and driving of data on both edges of the clock.

Off-chip memory.

Connected via memory controllers.

* + - 1. Key Features
         1. Volatile - data is erased when power is off.
    1. ROM - Read Only Memory
       1. Non - Volatile
          1. Data is preserved without power.
       2. Read - Only
          1. Contents cannot be modified.
       3. BootCode is written here - BootROM.
       4. Speed is slower than RAM.
       5. Flash - for IOTs, SoCs, etc.
  1. **I/O Ports - input output ports.**  
     1. In an SoC (System-on-Chip), I/O ports (Input/Output ports) are the interfaces that connect the digital core of the SoC to the outside world (other chips, peripherals, sensors, actuators, memory, or user devices).
        1. They are essential because an SoC by itself is just logic + memory; it needs I/O to communicate with external devices.
     2. Connect the SoC to other parts or devices, like a camera, USB, or even your headphones.
     3. These ports let the SoC send and receive data externally.
     4. Types of I/O in SoC
        1. General Purpose I/O (GPIO)
           1. Simple digital input or output pins.
           2. Configurable as input (read) or output (drive).
           3. Used for LEDs, switches, basic device control.
        2. Serial Communication Ports
           1. UART (Universal Asynchronous Receiver/Transmitter) → serial data transfer.
           2. SPI (Serial Peripheral Interface) → high-speed serial bus for sensors, flash, etc.
           3. I²C (Inter-Integrated Circuit) → low-speed bus for sensors, RTC, etc.
        3. Parallel Communication Ports
           1. Memory interfaces (e.g., parallel SRAM/Flash bus).
           2. Parallel display interfaces.

### High-Speed I/O

### USB → data transfer and device connectivity.

### PCIe → high-speed interconnect for GPUs, storage, accelerators.

### Ethernet → networking.

### MIPI (CSI/DSI, I3C) → camera and display connections.

### Analog I/O

### ADC inputs → for sensors, microphones, analog signals.

### DAC outputs → for audio, actuators.

### Analog pins for RF or mixed-signal circuits.

### Memory I/O

### DDR/LPDDR interfaces for external RAM.

### eMMC, SDIO for storage.

### How they work.

### Each I/O pin is controlled by an **I/O cell** in the **pad ring** of the chip.

### Functions of I/O cells:

### **Level shifting** (match internal core voltage, e.g., 1.0V, to external I/O voltage, e.g., 1.8V/3.3V).

### **ESD protection** (to prevent damage from static electricity).

### **Drive strength control** (strong enough to drive external loads).

### **Pull-up/pull-down resistors** (to define logic levels when not driven).

### I/O pins are usually **multiplexed** (same pin can act as GPIO, UART, SPI, etc., depending on configuration).

* 1. **Graphics Processing Unit (GPU):**  
     1. In an SoC (System-on-Chip), the GPU (Graphics Processing Unit) is a specialized processor designed to handle **parallel, high-throughput computations**, primarily for **graphics rendering** and increasingly for **AI/ML and compute workloads**.
     2. Why a GPU in SoC?
        1. CPUs are optimized for **sequential, general-purpose tasks.**
        2. GPUs excel at **parallel tasks** like pixel shading, matrix multiplications, and vector operations.
        3. In SoCs (smartphones, tablets, automotive, IoT, embedded systems), the GPU is integrated to provide:
        4. **Graphics acceleration** (UI, gaming, video).
        5. **Compute-acceleration** (AI inference, image/video processing).
     3. Functions of GPU in SoC.
        1. **Graphics Rendering**
           1. 2D and 3D graphics for displays (smartphones, TVs, automotive dashboards).
           2. Implements APIs like **OpenGL ES, Vulkan, DirectX**.
        2. **Video Processing**
           1. Sometimes GPU assists with video decoding/encoding (though often a separate VPU exists).
        3. **AI / ML Acceleration**
           1. Modern GPUs in SoCs (e.g., Qualcomm Adreno, ARM Mali, Apple GPU) accelerate **neural networks**.
           2. Parallel execution of **matrix operations (e.g., convolution)**.
        4. **GPGPU (General-Purpose GPU)**
           1. Beyond graphics → used for **signal processing, simulations, crypto, vision processing**.
     4. Responsible for creating visuals on your screen.
     5. Examples:
        1. **Qualcomm Snapdragon** → **Adreno GPU**
        2. **Apple A-series / M-series** → Custom Apple GPU
        3. **Samsung Exynos** → ARM Mali GPU or AMD RDNA GPU (newer ones)
        4. **MediaTek Dimensity** → ARM Mali GPU
        5. **NVIDIA Tegra (automotive, Jetson)** → NVIDIA GPU cores
     6. Summary
        1. **GPU in SoC = parallel processor for graphics + compute**.
        2. It enables **gaming, UI, video, and AI acceleration**.
        3. Modern SoCs use GPUs not just for rendering but also as **mini accelerators for ML and high-performance tasks**.

* 1. **Digital Processing Unit (DSP):**  
     1. In an **SoC (System-on-Chip)**, a **DSP (Digital Signal Processor)** is a specialized processor core designed to handle **mathematical operations on streams of data** efficiently — especially operations like **multiplication, addition, filtering, and Fourier transforms**.
     2. Why DSP in SoC?
        1. CPUs are general-purpose but not optimized for heavy repetitive math.
        2. DSPs are optimized for **real-time signal processing**, where speed and low power are critical.
        3. Widely used in **audio, video, communications, radar, and AI/ML** inside SoCs.
     3. Features of DSP in SoC
        1. **Special Instruction Set**
           1. Multiply-Accumulate (MAC) operations in a single cycle.
           2. SIMD (Single Instruction, Multiple Data) for parallel processing.
        2. **Hardware Optimizations**
           1. **Harvard architecture** (separate program & data memories for speed).
           2. **Circular buffers** for real-time data streams.
           3. **Zero-overhead looping** (fast repetitive operations).
        3. **Low Power**
           1. Efficient for embedded and portable devices.
     4. Functions of DSP in SoC
        1. **Audio Processing**
           1. Noise cancellation, echo suppression, audio compression (MP3, AAC).
           2. Voice assistants, speech recognition.
        2. **Image & Video Processing**
           1. JPEG/MPEG/H.264 compression, filtering, scaling.
           2. Augmented reality and computer vision.
        3. **Communications**
           1. Modulation/demodulation (QAM, OFDM).
           2. Channel equalization, error correction.
           3. Baseband processing in 4G/5G modems.
        4. **AI/ML Acceleration**
           1. Vector/matrix operations for lightweight neural networks.
     5. Examples
        1. **Qualcomm Hexagon DSP** → audio, AI, modem tasks.
        2. **TI C6000 DSP cores** → industrial/automotive SoCs.
        3. **ARM Cortex-M4/M7 (with DSP extensions)** → microcontrollers.
        4. **Apple A-series** → uses DSP blocks for image/audio processing.
     6. Summary
        1. **DSP in SoC = a math engine for real-time digital signal processing**.
        2. Handles **audio, video, comms, and AI tasks** efficiently.
        3. Complements CPU (control) and GPU (parallel compute).
  2. **Power Management Unit - [PMU]**
     1. In an **SoC (System-on-Chip)**, the **PMU (Power Management Unit)** is the subsystem responsible for **managing and controlling power consumption, voltage regulation, and power modes** across the chip.
     2. Why is PMU important in SoC?  
        1. Modern SoCs integrate **CPU, GPU, DSP, memory controllers, I/O, analog blocks** → all of which have different power requirements.
        2. Power efficiency is critical in **mobile devices, IoT, automotive, and high-performance processors**.
        3. The PMU ensures the SoC gets the **right voltage, at the right time, with minimal power wastage**.
     3. Functions of PMU in SoC
        1. **Voltage Regulation**
           1. Controls **on-chip voltage regulators / DC-DC converters / LDOs**.
           2. Supplies different voltage domains (e.g., CPU @ 1.0V, I/O @ 1.8V, SRAM @ 0.9V).
        2. **Dynamic Voltage and Frequency Scaling (DVFS)**
           1. Adjusts **voltage + clock frequency** depending on workload.
           2. Example: Lower CPU frequency in idle → saves battery.
        3. **Power Gating**
           1. Shuts off unused blocks (GPU, DSP, cores) to save leakage power.
           2. Essential in **low-power SoCs**.
        4. **Clock Gating**
           1. Disables clock signals to idle modules, reducing dynamic power.
           2. **Battery & Thermal Management**
           3. Monitors battery level, charging, temperature sensors.
           4. Prevents overheating and overcurrent damage.
        5. **Sleep / Wake-up Control**
           1. Manages **low-power states** (sleep, deep sleep, hibernate).
           2. Controls wake-up events (interrupts, timers, external signals).

## **PMU in SoC Architecture**

## Usually implemented as a **dedicated microcontroller or FSM** inside the SoC.

## Interacts with:

## **CPU/GPU/DSP cores** (to manage DVFS).

## **Clock & reset controller** (for clock gating).

## **External PMIC (Power Management IC)** in mobile devices.

* + 1. **Summary**
       1. **PMU “the power brain of an SoC”**.
       2. Handles **voltage scaling, frequency scaling, power/clock gating, and thermal/battery management**.
       3. Critical for **energy efficiency, battery life, and thermal stability** in modern SoCs.
  1. **Special Features**
     1. **Wireless Connectivity** 
        1. **Wi-Fi** (802.11 a/b/g/n/ac/ax) → enables networking and internet access.
        2. **Bluetooth** (Classic, BLE) → short-range wireless for audio, wearables, IoT.
        3. **Cellular modem** (2G/3G/4G/5G) → integrated in mobile SoCs.
        4. **GNSS** (GPS, GLONASS, Galileo, BeiDou) → location tracking.

### **Security Modules**

* + - 1. **Hardware Security Engine (HSE)** → accelerates cryptography (AES, RSA, SHA).
      2. **Trusted Execution Environment (TEE)** → secure area for running sensitive code.
      3. **Secure Boot & Key Storage** → ensures only trusted firmware runs.
      4. **DRM & Content Protection** → used in streaming, payments, biometrics.

### **Multimedia Engines**

* + - 1. **ISP (Image Signal Processor)** → camera processing (noise reduction, HDR, face detection).
      2. **VPU (Video Processing Unit)** → hardware video decode/encode (H.264, H.265, AV1).
      3. **Audio DSP** → noise cancellation, voice recognition, spatial sound.

### **AI / ML Accelerators**

* + - 1. Dedicated **Neural Processing Units (NPU/TPU)** for deep learning inference.
      2. Faster and more power-efficient than using CPU/GPU for AI tasks.
    1. **Peripheral Interfaces**
       1. **USB, PCIe, MIPI (CSI/DSI/I3C), SDIO, Ethernet**.
       2. Enable SoC to connect with external devices (storage, displays, cameras, sensors).
    2. **Power & System Management**
       1. **PMU (Power Management Unit)** for DVFS, clock gating.
       2. **Thermal sensors** for safe operation.**B**
       3. **Battery charging circuits** (common in mobile SoCs).
    3. **Summary**
       1. In an SoC, special features like Wi-Fi, Bluetooth, security modules, AI accelerators, and multimedia engines make the chip suitable for specific applications (smartphones, IoT, automotive, etc.). They vary depending on the target use-case of the SoC.
  1. **Challenges with SoC’s.**
     1. **Complex Design:** Creating an SoC is complicated. Combining multiple functions in one small space requires advanced design skills.
     2. **Heat Issues:** Packing many components together can lead to overheating. SoCs need cooling solutions to work well over time.
     3. **Less Flexibility:** Once an SoC is designed, it’s hard to change. This is because each SoC is built for specific tasks or devices.
  2. **Types of SoCs.**
     1. **Microcontroller-based SoC:** These SoCs are built around a microcontroller, designed for simple control tasks in everyday devices. Known for their low power usage and efficiency, they’re perfect for applications like home appliances, car systems, and IoT devices, where processing needs are minimal, and power savings are essential.
     2. **Microprocessor-based SoC:** This type features a microprocessor, which can handle more demanding tasks and run operating systems. Commonly used in smartphones and tablets, microprocessor-based SoCs manage multiple tasks and support complex applications, providing the higher processing power necessary for interactive and data-intensive applications.
     3. **Application-Specific SoC:** Custom-designed for specific, high-performance tasks, these SoCs excel in areas like graphics processing, network management, and multimedia applications. Optimized for speed and efficiency in their designated roles, they’re often used in graphics cards, AI hardware, and specialized industrial or financial systems that require precise, fast processing.

1. **VSDBabySoC**
   1. **RVMYTH:** A simple RISC-V-based CPU core.
      1. Pre loaded with:
         1. Instructions
         2. Reg Values
   2. **PLL:** An 8x Phase-Locked Loop for stable clock generation.
      1. Pre Initialized with CLK Values
   3. **DAC:** A 10-bit Digital-to-Analog Converter for interfacing with analog devices.