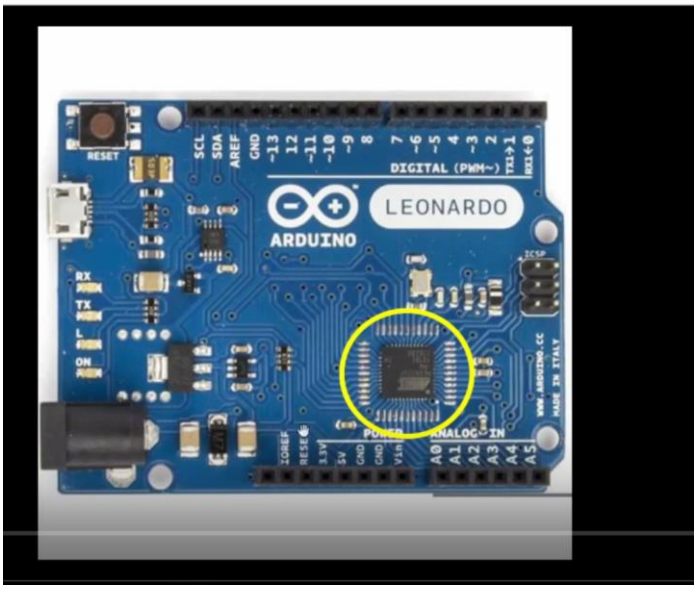
DAY-1 INCEPTION OF OPEN-SOURCE EDA, OPEN LANE & SKY130 PDK

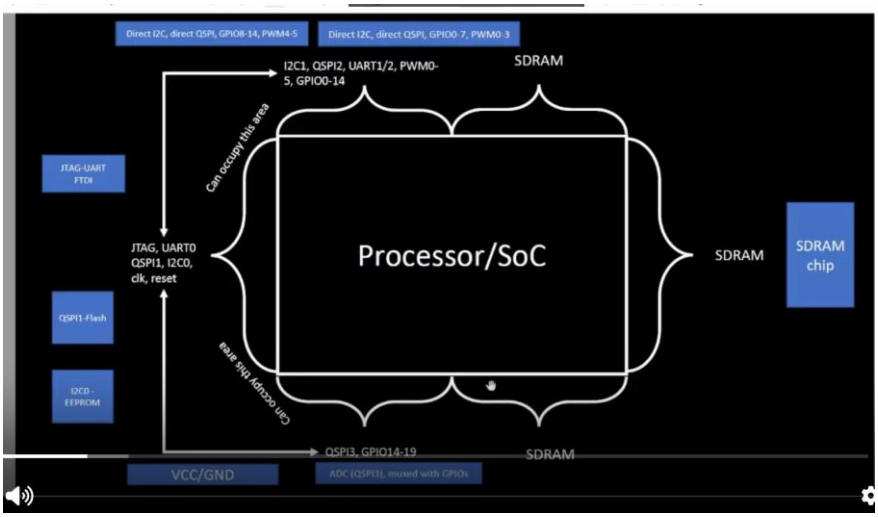
1. HOW TO TALK TO COMPUTERS?

L1: Intro to QFN-48, Package, chip, pads, core, die and IP's

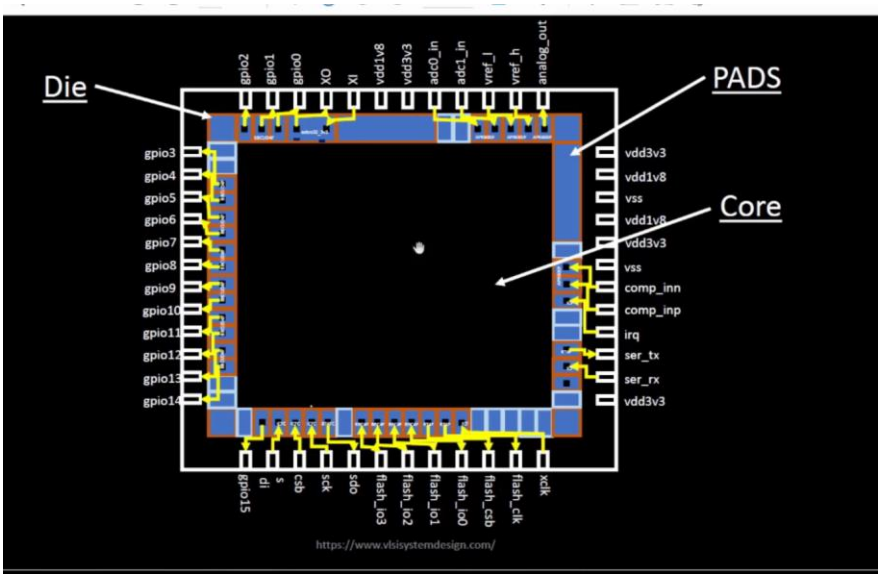
- **Arduino Board:** The ATmega32u4 microcontroller on the Arduino Leonardo comes in a QFN-48 package. QFN-48 stands for Quad Flat No-lead, 48 pins. It is a type of surface-mount integrated circuit (IC) package.



- **Block diagram of the SOC**

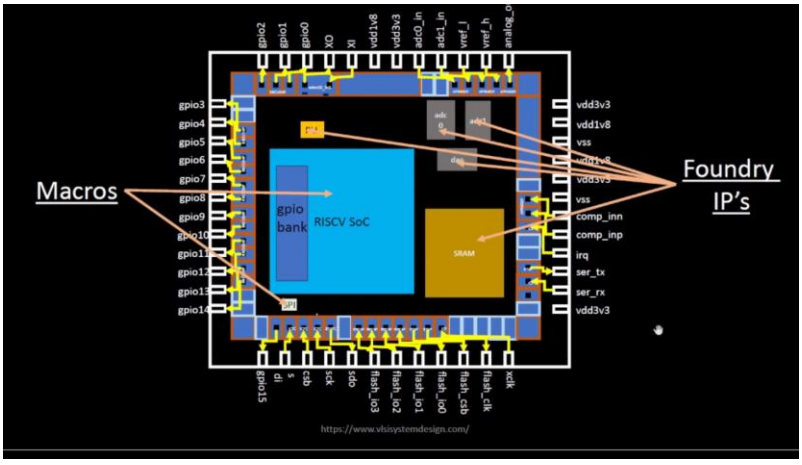


- **How the pads, core and die are placed and connected inside the chip.**



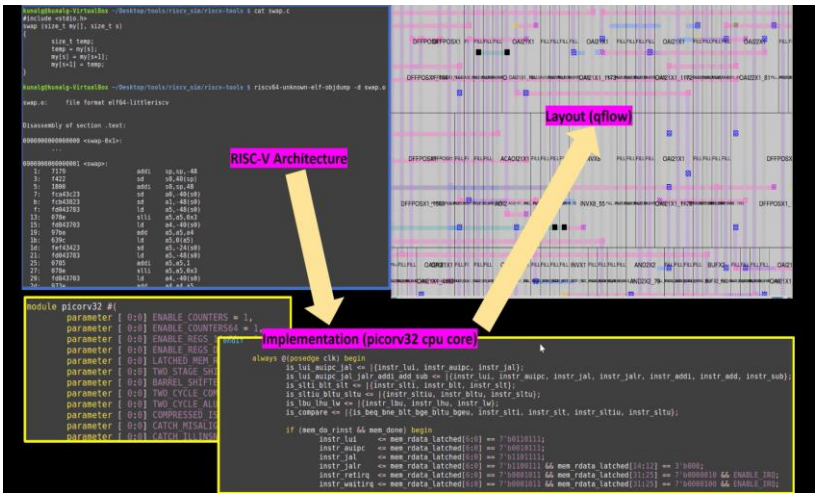
- **RISC -V Processor chip diagram**

1) **Foundry IP's:** IP stands for Intellectual property blocks that are provided by semiconductor foundries and are used to implement or support parts of the processor or SoC. Foundry IPs are used alongside the RISC-V core (which is typically RTL code or soft IP) to build a complete chip.
2) **Macros:** Macros are built by pure digital logic like and gate, mux flipflops etc.,



L2: Intro to RISC-V

- **The flow of RISC-V architecture is as below.**
  In the given example a C program of swap two numbers is given this is converted to an Assembly level language and given to the layout in the form of GDS II file which represents the layout data.
  Here picorv32 CPU core acts as an interface between the RISC-V architecture and layout. picorv32 adheres to the ISA of RISC-V and hence is used.



L3: From Software applications to Hardware

- **How applications of a software are mapped to hardware?**

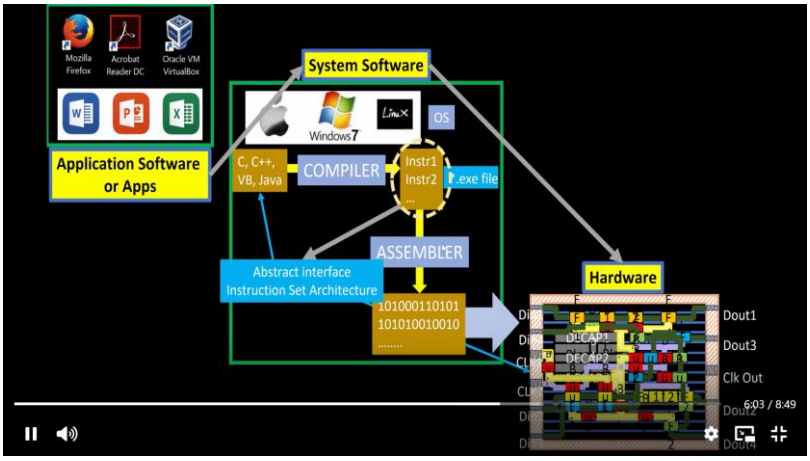
  The System software helps the apps to connect to hardware through OS, compiler and assembler.

  OS: Handles the memory organisation, tasks priority etc.,

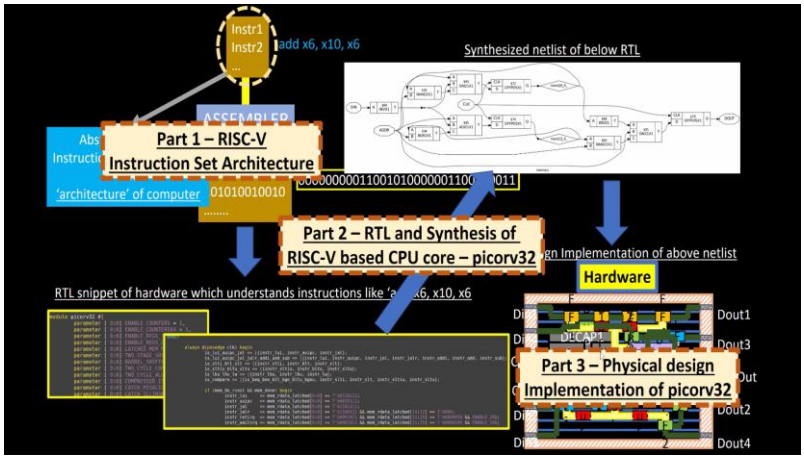
  COMPLIER: A high-level language like C, C++, Java etc., is given to complier which converts into Instructions according to the ISA SOC (here its RISC-V processor) and then is given in a .exe file.

  ASSEMBLER: Assembler converts this to a hardware readable language i.e., in the form of 0's and 1's also known as binary language and then is fed to the hardware to perform its function.

  The below Is the image of the flow from software to hardware.



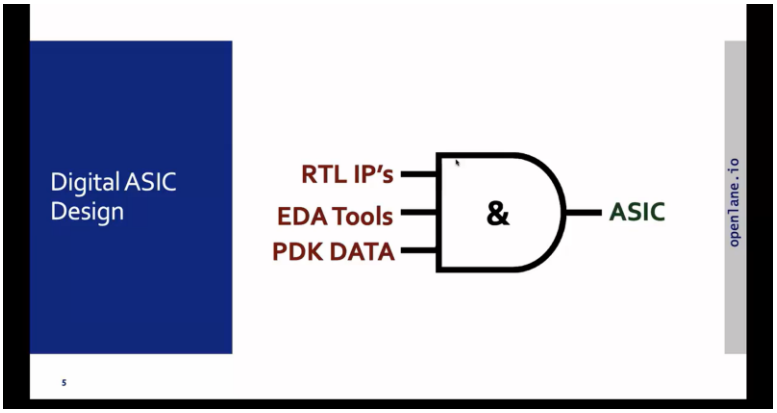
Further more we need an RTL which understands the instructions given by assembler for the synthesis and to later perform physical design of the netlist. An example snapshot is shown below.



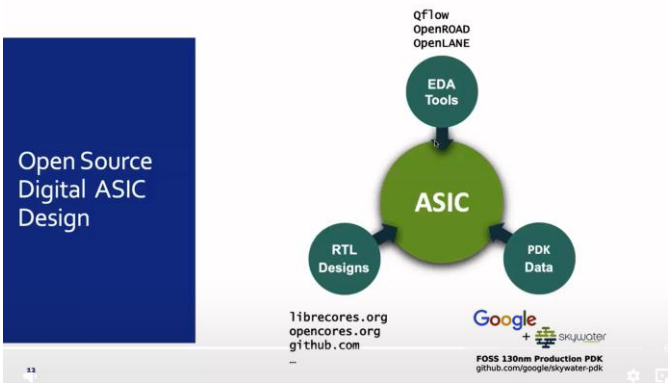
2. SOC DESIGN AND OPENLANE

L1: Introduction to all components of open-source digital ASIC design

- **The components required for an open source to perform ASIC design is as below.**

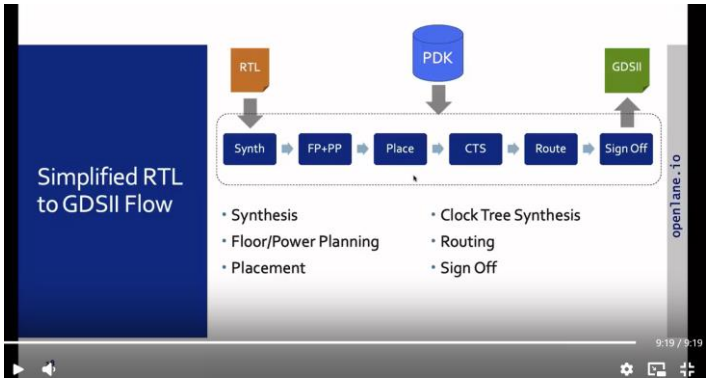


- **The components mentioned in the below picture are all certified open-source which can be used by all to implement an ASIC design.**



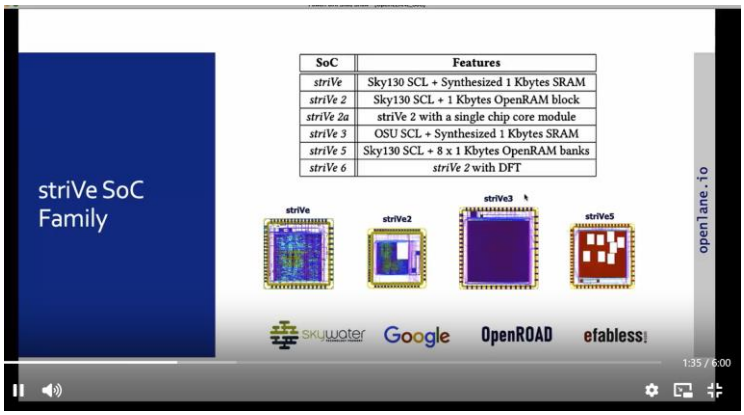
L2: Simplified RTL2GDS flow

- **The flow** for implementing any design starts from RTL design to GDS II file which is needed for the physical design of the hardware ie., the layout of the hardware.
  At Each step the required is done and forwarded to the next step.



L3: Introduction to OpenLane and strive chipsets

- **StriVe:** is a family of open everything SoCs ie., open PDK, open EDA and open RTL. The picture shows different Soc's in StriVe family.



- **OpenLANE ASIC Flow**: The main goal Produce a clean GDSII with no human intervention (no-human-in-the-loop) ie., without LVS Violations, DRC Violations and with minimum Timing Violations.
- **OpenLane features:** Tuned for skywater130nm open pdk, Containerized, it can be used to harden Macros and chips, it has 2 modes of operations (Autonomous and Interactive) and many more.

L4: Introduction to OpenLane detailed ASIC flow

- **The below is the flow** of OpenLane from RTL to GDS II and SKY130 PDK is function that helps to run the flow.