RISC V theory

Day – 1

* Assembly language programme is converted to machine learning programme
* Hardware interface language
* RISC V architecture is implemented using RTL and then it is the RTL to GDS II flow to obtain lay out
* How do App’s run on hardware?
* System software consists of OS ,Compiler and assembler
* OS handles IO operations ,allocates memory and low level system functions , it also converts it to assembly language programme
* Output of the OS may be C, C++ , java languages
* The respective compilers receive the Output and convert it into instructions
* The syntax of the instructions depends on the type of hardware
* The .exe file consists of such instructions
* Assembler converts the instructions into respective binary language /machine language programme
* Hardware generates the output based on the HDL (RISC V architecture)
* Hexadecimal numbers are the output of the assembler
* The instructions are the abstract interface between the C language and the hardware
* This is called ISA (instruction set architecture - RISC V architecture)
* Pseudo instructions
* mv,li,ret
* base integer instructions (RV64i)
* lui,addi, sd, aupc, jalr, lw,ld ,divw, mul
* rv32 or rv 64 based on bit
* multiply extension (RV64m)
* mw, dw
* RV64im cpu core – implements both base integer instructions as well as multiply extensions
* Floating point numbers – f
* They are single and double precision extensions (RV64F) & (RV64D)
* A cpu that implements all the above functions (RV64IMFD)
* Application binary interface (ABI) – Written address is associated with registers which are part of RISC V ISA
* Memory allocation stack pointer – data is transferred between stack pointers, memory & registers
* An entire 64 bit number is called double word, where as 32 bit is called word
* The last digit is called least significant bit where as first digit is the most significant digit
* A group of 8 bits is called a byte
* 4 bytes form a word
* 8 bytes form a double word
* 8 bits → 1 byte → 4 bytes → 1word → 2 words or 8 bytes → double word
* Total number of patterns represented by RV 64 is 264
* Lowest number all 0’s
* Highest number all 1’s
* Overflow flag
* Negative number representation (Twos complement representation )
* Getting binary format of the number and inverting all 0 to 1 and vise versa
* Then add one to the final output
* MSB for positive numbers is 0 and negative numbers is 1
* The range of negative numbers that can be represented is -1 to -263

Day -2

Application binary interface

* Interface – apprarance & functionality,
  + Codes have to be written
* User ISA available to application programmer
* User and system ISA available to operating system
* Programmer has access to system calls and hardware
* These system calls are called ABI
* ABI functions through Registers
* XLEN defines the width of the registers
* Registers
* Why are there only 32 registers in RSC V architecture
* Two ways to load – in a 64 bit register
* Load directly
* Memory
* Memory addresses a byte
* Little Endian memory system which starts with LSB byte
* Big Endian memory system which starts with MSB byte
* Address of first double word is m0
* Address of a 2nd double word id m8
* Command ld for load double word
* Instruction size is 32
* Instructions operating on registers are called R type
* Instructions operating on registers and Immediate called I type
* Instructions operating on source registers and Immediate called S type
* All registers are 5 bits
* There fore all registers are 25 ,32 bits

By archana

The prev labs till the error I faced in day 5

<http://makerchip.com/sandbox/04xfJhB2G/048hBQ0#>

grateful to KUNAL SIR for the opportunity

and to steeve hoover for teaching a lot of interesting things