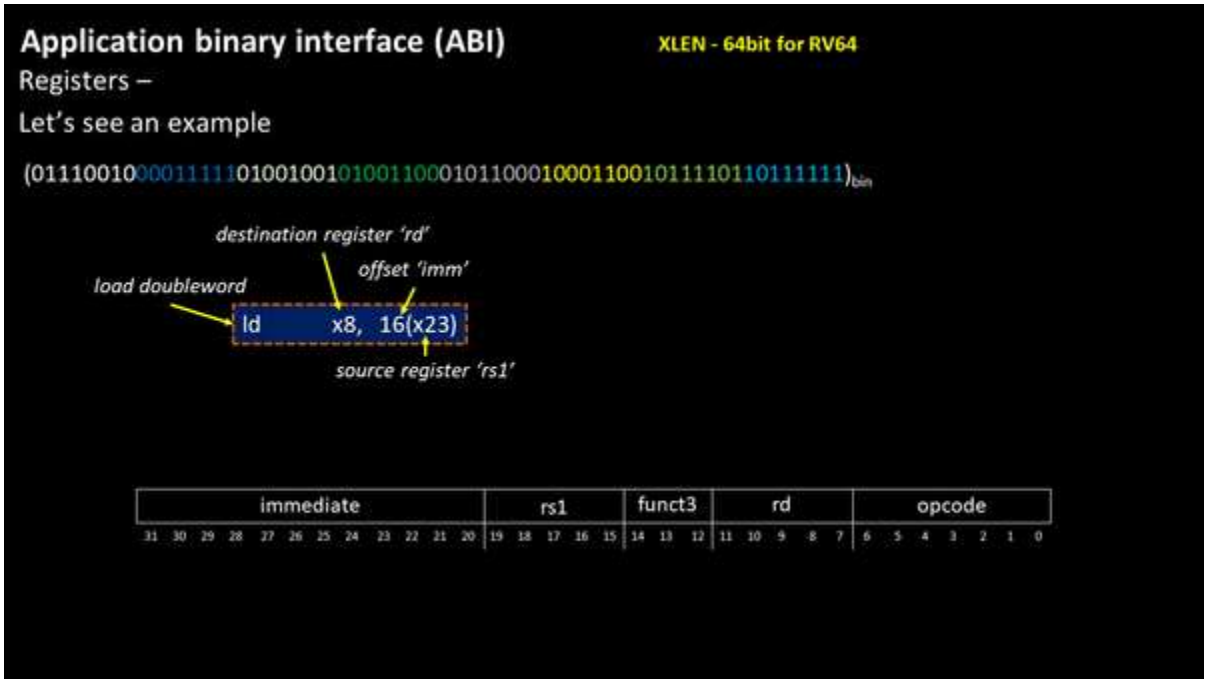
Why RISC-V architecture has 32 registers?

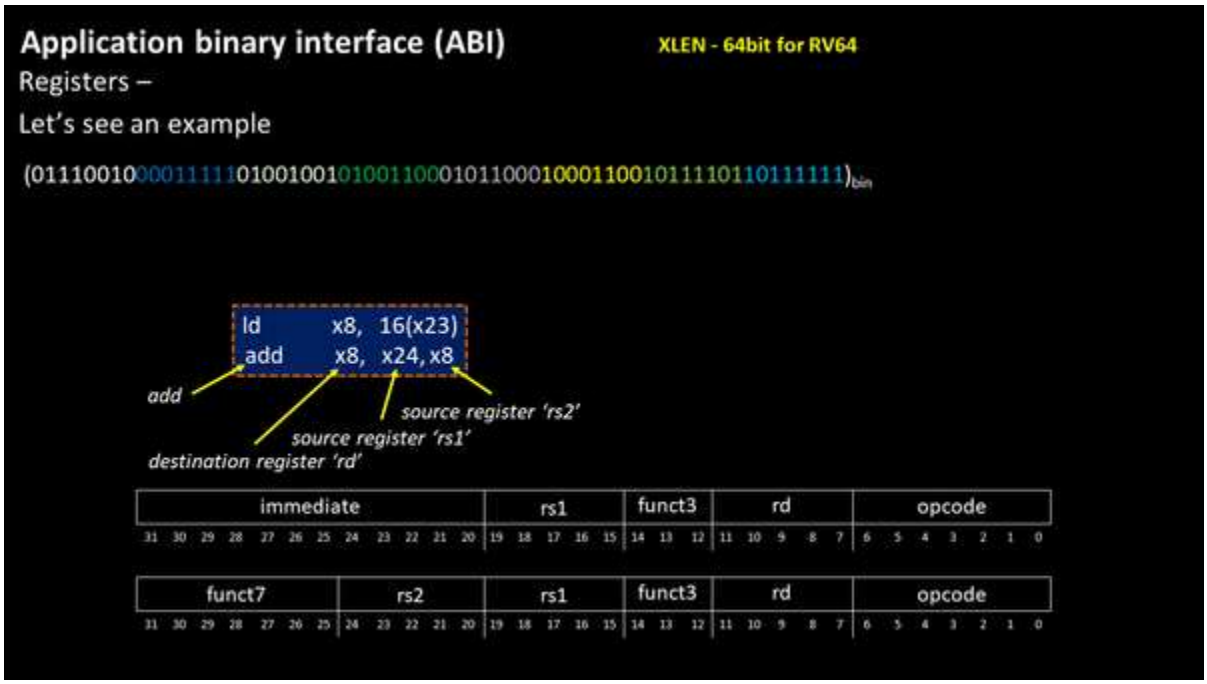
Kunal Ghosh

Hi
Many such architectural questions has been solved in my launched online course on RISC-V ISA.
Have you ever thought, why RISC-V instruction sets, or for that matter, most instruction sets today have 16 or 32 general purpose registers? Something to think about
Let's look at an example *load doubleword* instruction below, which loads data into x8 register from memory, whose base address is present in register x23 and offset is '16'. The way a computer sees this instruction is through a 32-bit binary pattern. (Details of the below command is covered in the course)



Now focus on 'rd' and 'rs1' which are called as *destination register* and *source register* respectively. Each of them has been strictly allotted 5-bits. Which means, to represent x8 for 'rd', the field will contain the pattern '01000'

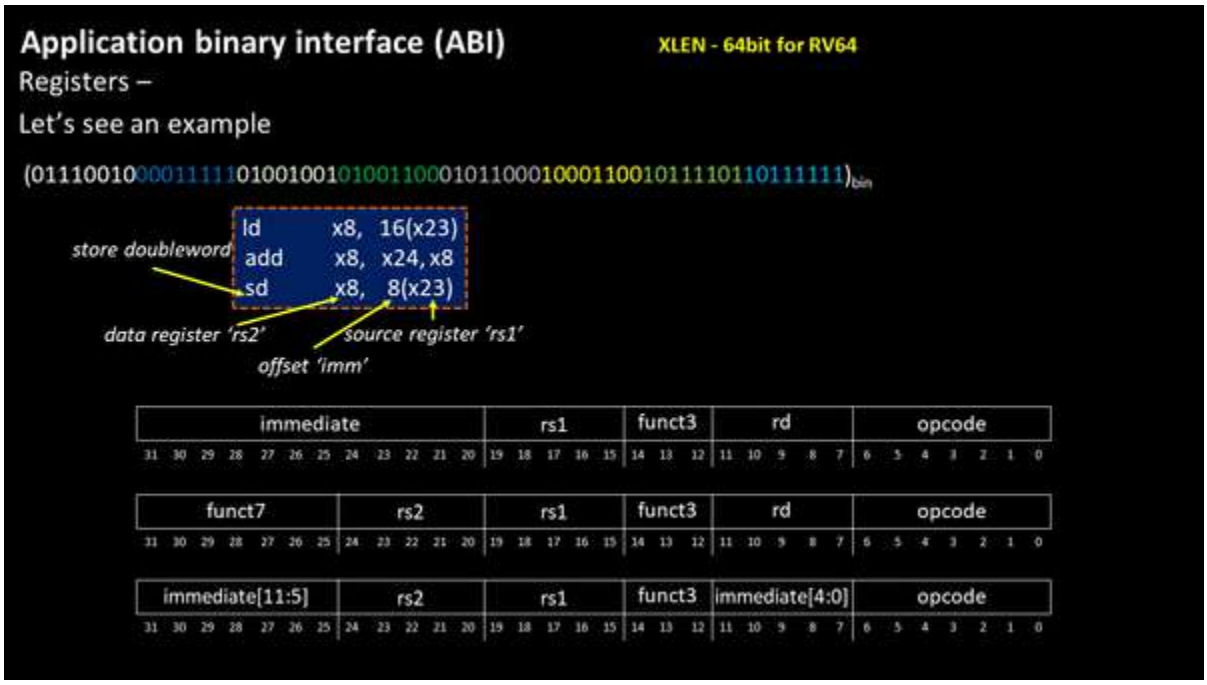
Another example, shown below, adds contents of x8 with data present in x24 and stores results back in x8. Here, we have 2 *source registers* x24 and x8, and one *destination register* x8



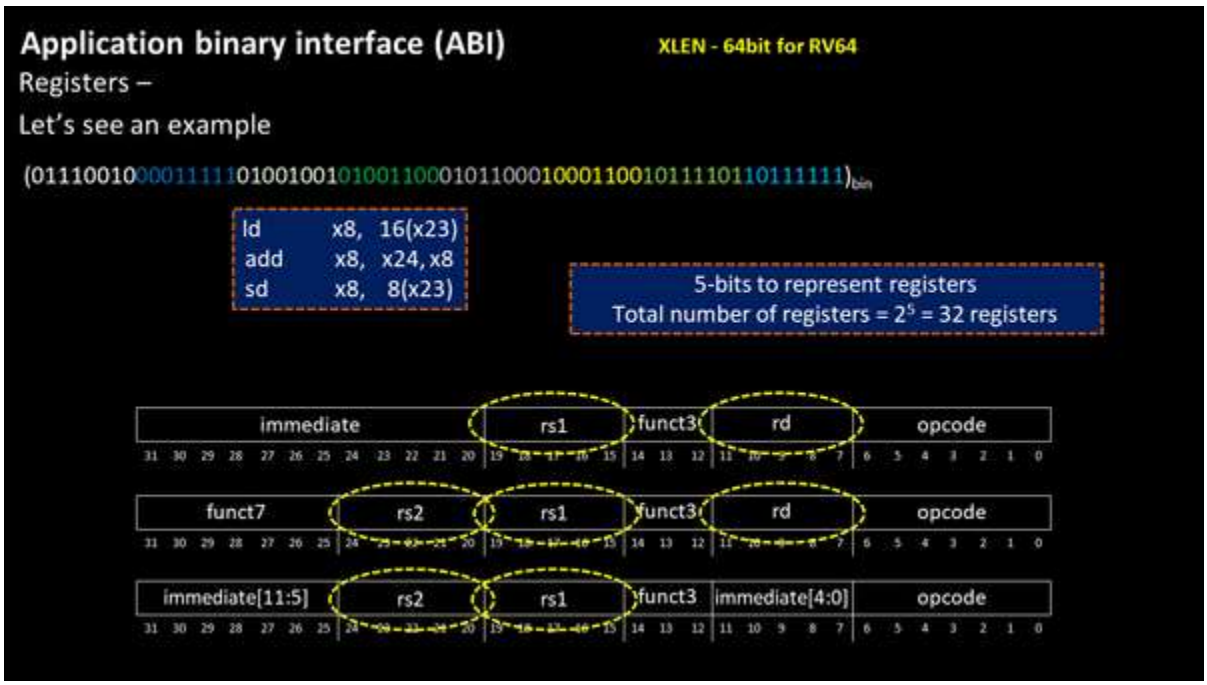
Again, focus at 'rd', 'rs1' and 'rs2'. They are all 5 bits, which means, to represent x24 in rs1, that field will contain '11000'

One is good, two is better, three is best…So let's consider third example below

In the example, the contents or 'x8' are stored in memory whose base address is present in register 'rs2' and offset is '8'



Once more, focus on 'rs2' and 'rs1'. They are again 5 bits. Practically, to keep design simple, all registers in a RISC-V architecture is represented by 5-bit binary pattern.



Now the calculation is easy. ***5-bits to represent registers, which means total number of registers is 2^5 = 32 registers***
"***Simplicity favors regularity and good design demands good compromises***" – Just as Prof. David Patterson mentions in his book called "*Computer Organization and Design*"

The good part has already been per-launched and best part is yet to come.

The reason we launched this course, is because, we were asked to come with a design, which we can take through the entire RTL-synthesis-PNR-STA tool chain and after a lot of thought, we believed that understanding a sophisticated RISC-V CPU core is the best way to learn how to write specs, how to implement and how to PNR…all in one…

Stay tuned, more exciting things yet to come…

Till then…happy learning…