Project Report (Part-02.1)

CSE332

SECTION: 9

Submitted to:

Syed Mahmud Husain (SMH2)

Department of Electrical & Computer Engineering

Submitted by:

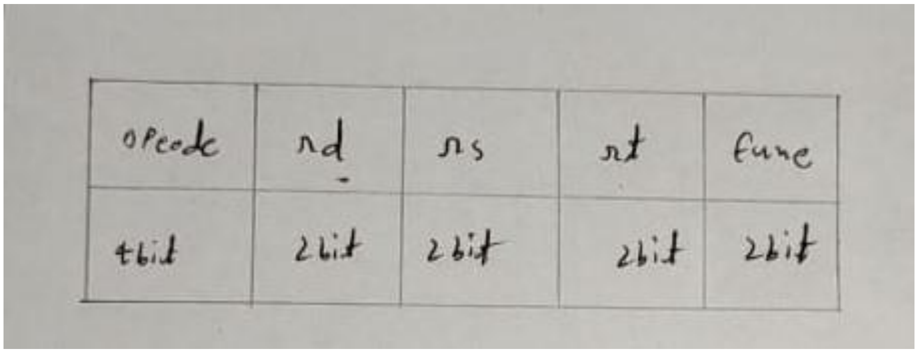
Muhammad Zawad Mahmud

ID: 1931401642

Data Table:

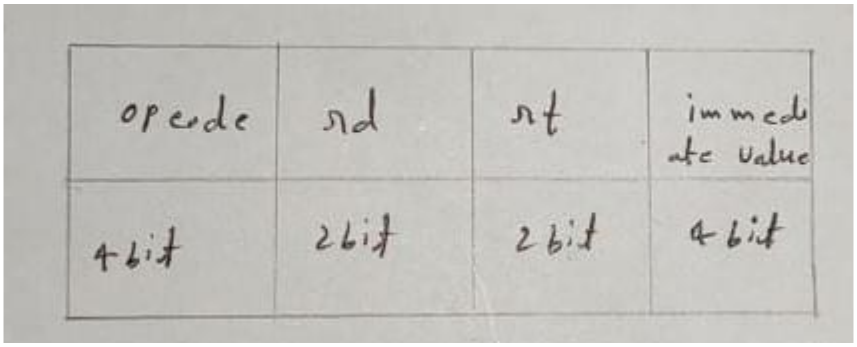
N=abs(my serial-32)=abs(30-32)=2
as N<10. So, N=2+10=12

R -Format:



oPcode	rd	rs	rt	func
4bit	2bit	2bit	2bit	2bit

I-Format:



opcode	rd	rt	immediate value
4bit	2bit	2bit	4bit

Result of simulation:

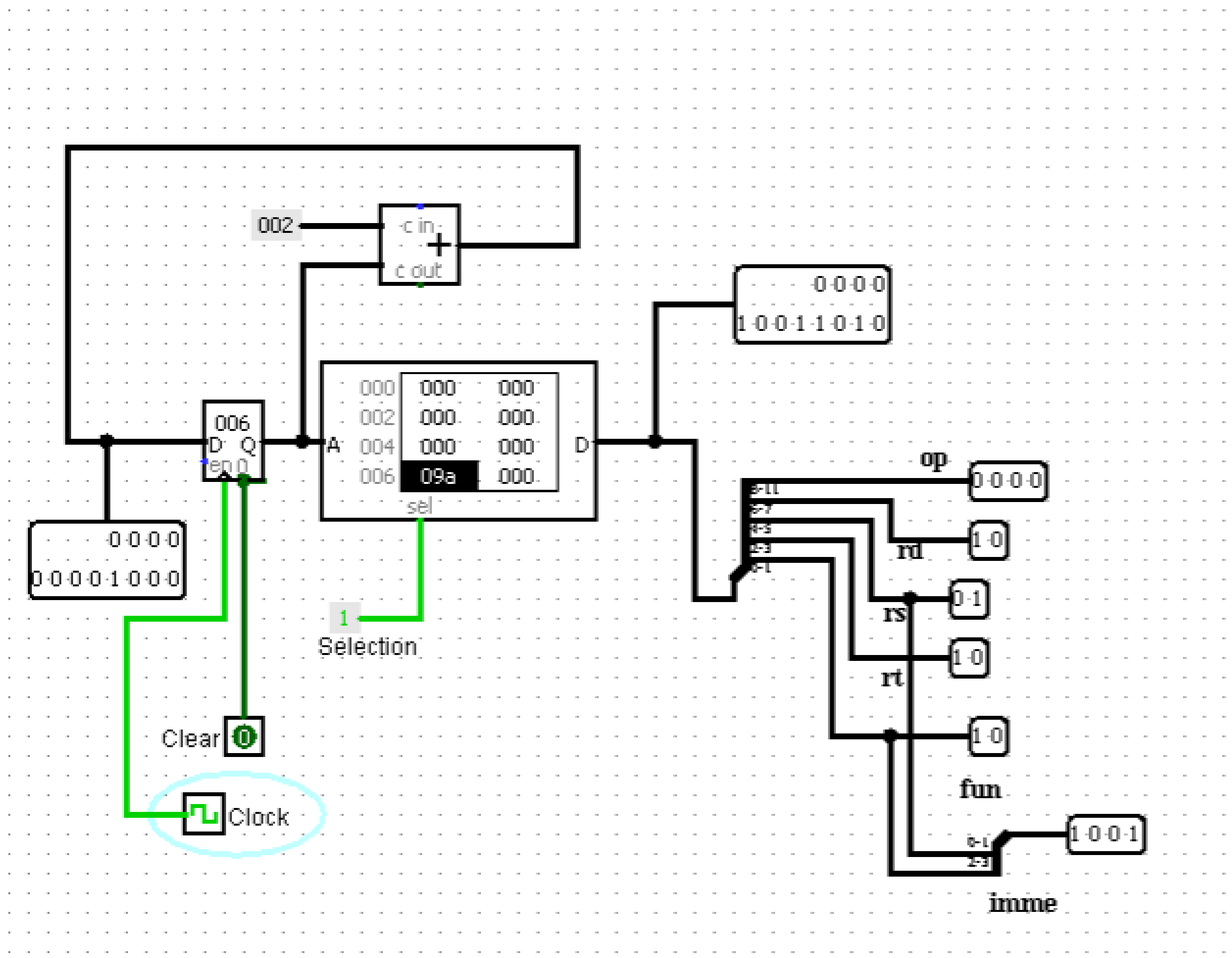


Figure: Data-path of instruction fetch

Discussion:

In this part 2 of my project, the objective was to design a full data-path of single cycle implementation of MIPS.

For hardware specification first I have to try to divide my bit in ISA format which is,
$N=2^n$
$12=2^n$
$n=\log12/\log2=3.58 \approx 4$
So that I need to give rs=rd=rt=4 which I can't give then I will be left with 0 bits but I'll need 2 bit each for op and function. So, I need to go for customized way. For functions, I have to give 2 bits as I need to perform 4 operations (load, store, add, sub). After giving 2 bit at func, I left with 10 bits. I can give 2 bits each for rs, rd & rt so I have given 2 bits for those and 4 bit to opcode.

Now in this part 2.1 of my project, I have designed data-path of instruction fetch. For this implementation, I needed program counter (PC). I have taken 12-bit rising edge register as PC. Then I needed an instruction memory. I have used rom as my instruction memory. I also needed a 12-bit adder. Firstly, I took a 12-bit register and an adder. I put register's output and 002 as adder's input as I'll need to add 2 while clock ticks. I connected a clock with my register and the input is adder's output. The clock will help to go to the next stage. Whenever the clock will tick, register will go to next stage which is the current stage+2. In memory each address can store 8 bits that's why for ISA, we increment by 4. Here as it is customized and it's 12 bit CPU so, my increment will be 2 otherwise there will be memory lose for two addresses. I also have a clear pin in my register which will take it to 0 stage asynchronously. I gave the register's output as my rom's address bit. The address bit and data bit of my rom both are 12 bits as I need to design a data-path for 12-bit CPU. I used a constant 1 as rom's selection pin as when the selection pin is 0, the rom is disabled. In rom, I have stored hexa-decimal numbers but with its data bits it will give binary form of the stored number as output. At last, I used a splitter to split the rom's output into opcode(4 bit), rd(2 bit), rs(2 bit), rt(2 bit), func(2 bit).

Then I used another splitter to add rs&func for the immediate value which I need for the I format.