# DP_Datapath_21363904

The timing diagrams for the Datapath are shown below. All numbers are in Hexadecimal notatation. The first three diagrams show each of the registers and temp registers being loaded with my Student ID – $n$ where $n$ is the number of the register being loaded (although for the temp registers, it's technically ID – 31 – $n$). The red lines in these diagrams mark the change in input at $DATAIN$ via $MuxD$ as each register is loaded.

The second three diagrams show each of the Datapath Operations in the order determined by the last digit of my Student ID (4), shown below. The red lines in these diagrams mark when the correct result appears at the output. The Destination Register (i.e. the register to which the result of the operations is written to) is selected by the last digit of my Student ID (4). Source Registers $A$ and $B$ (i.e. the registers from which the values to be operated on are read from) are selected by the last digit of my Student ID plus 5 and 15 respectively (9 and 19).

Each of the 15 operations are performed in the order specified in the Simulation Instructions for a Student Number with the last digit 4, the correct results of which are displayed in the waveform. The order of instructions is as follows:

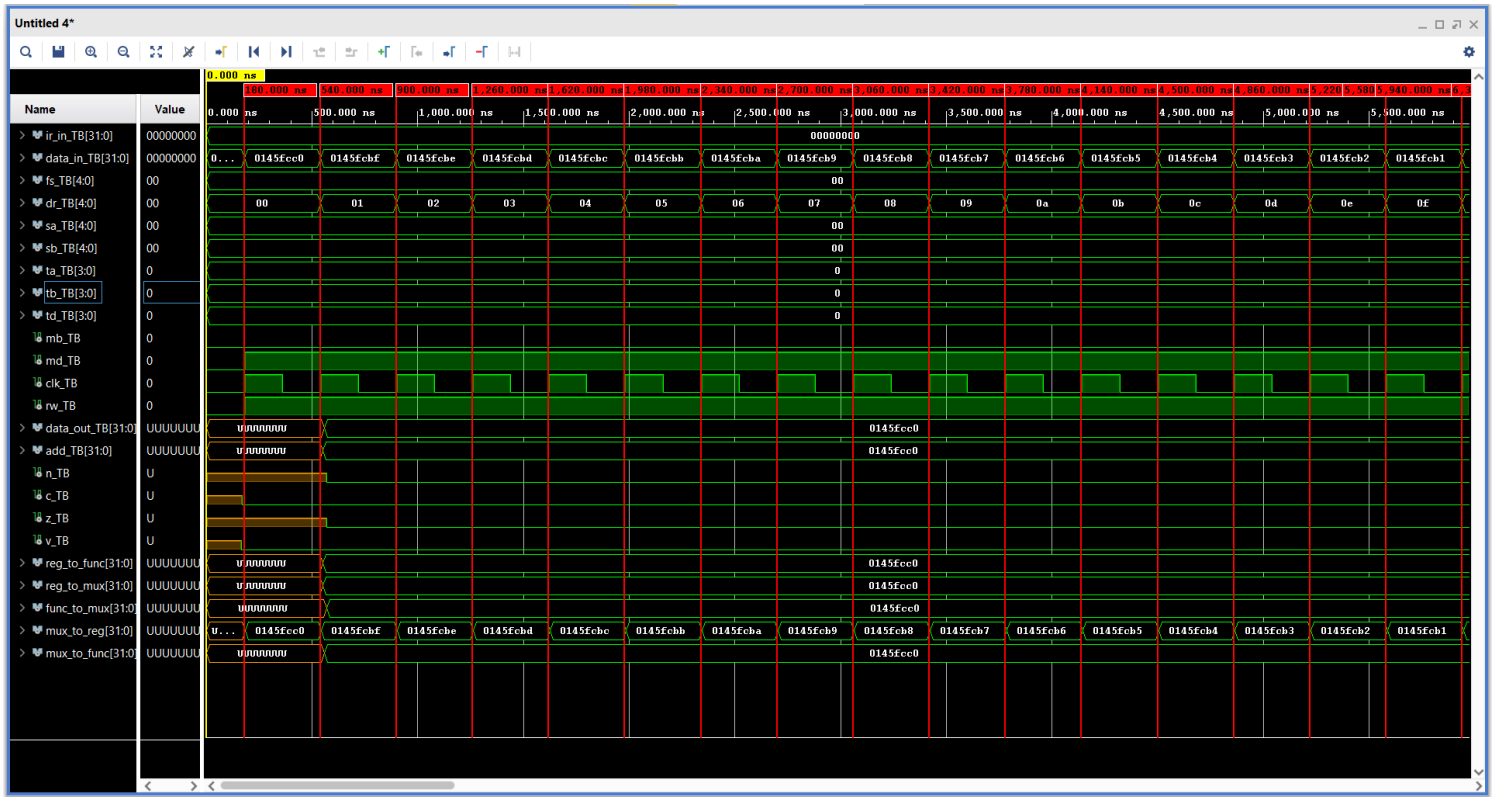
$$A + NOT(B) + 1, \quad NOT(A), \quad A + NOT(B), \quad A\,(00111),$$
$$A + B + 1, \quad B \gg, \quad A + B, \quad B \ll, \quad A\,(00000), \quad B, \quad A + 1,$$
$$A\ XOR\ B, \quad A\ OR\ B, \quad A - 1, \quad A\ AND\ B$$

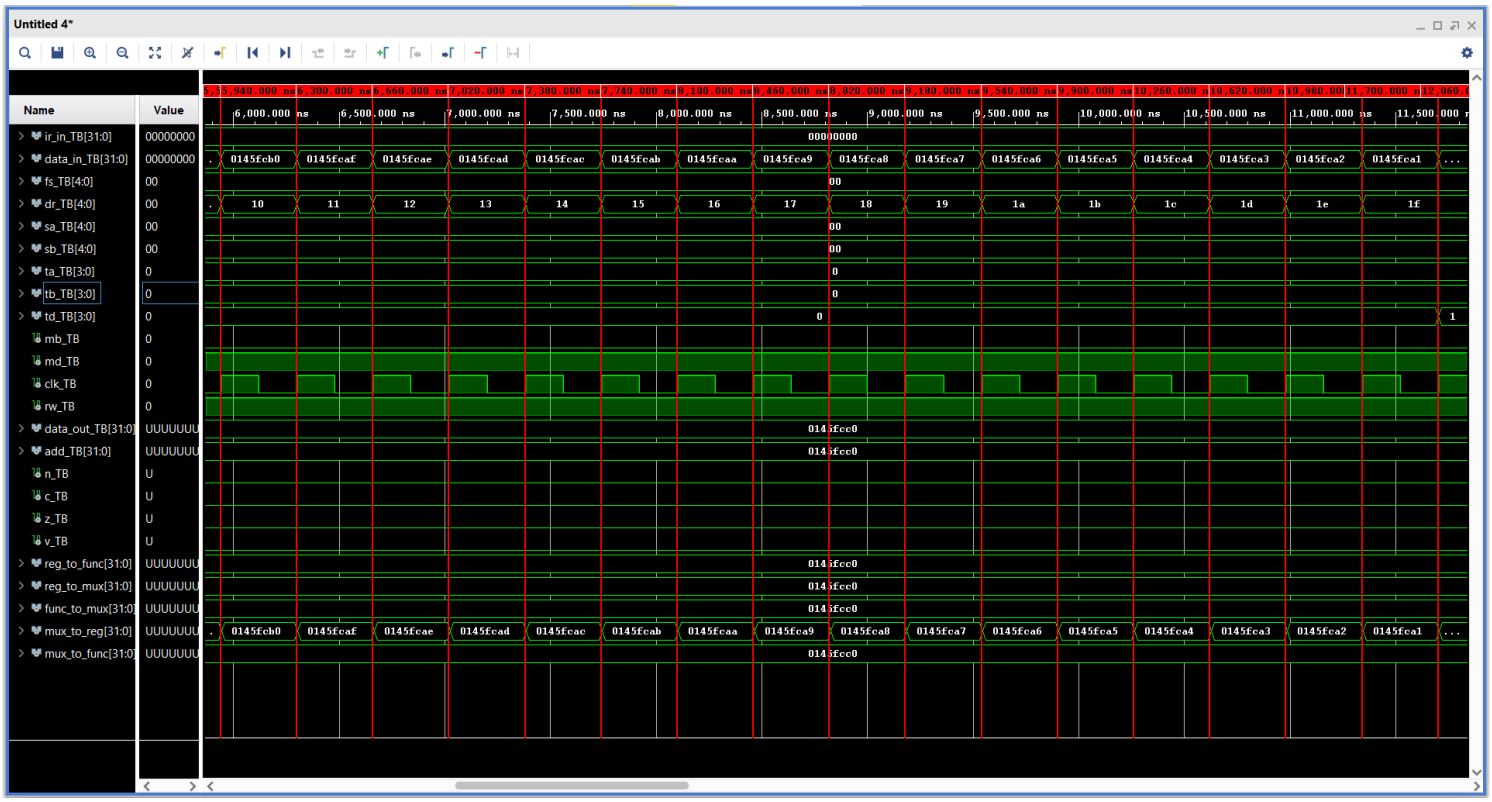
A further 10 operations are performed with $MuxB$ selecting 'Constant in' which is provided with my Student ID as requested. Only the operations involving a $B$ input are performed, in the same order as above, shown below.

$$A + NOT(B) + 1, \quad A + NOT(B), \quad A + B + 1, \quad B \gg,$$
$$A + B, \quad B \ll, \quad B, \quad A\ XOR\ B, \quad A\ OR\ B, \quad A\ AND\ B$$

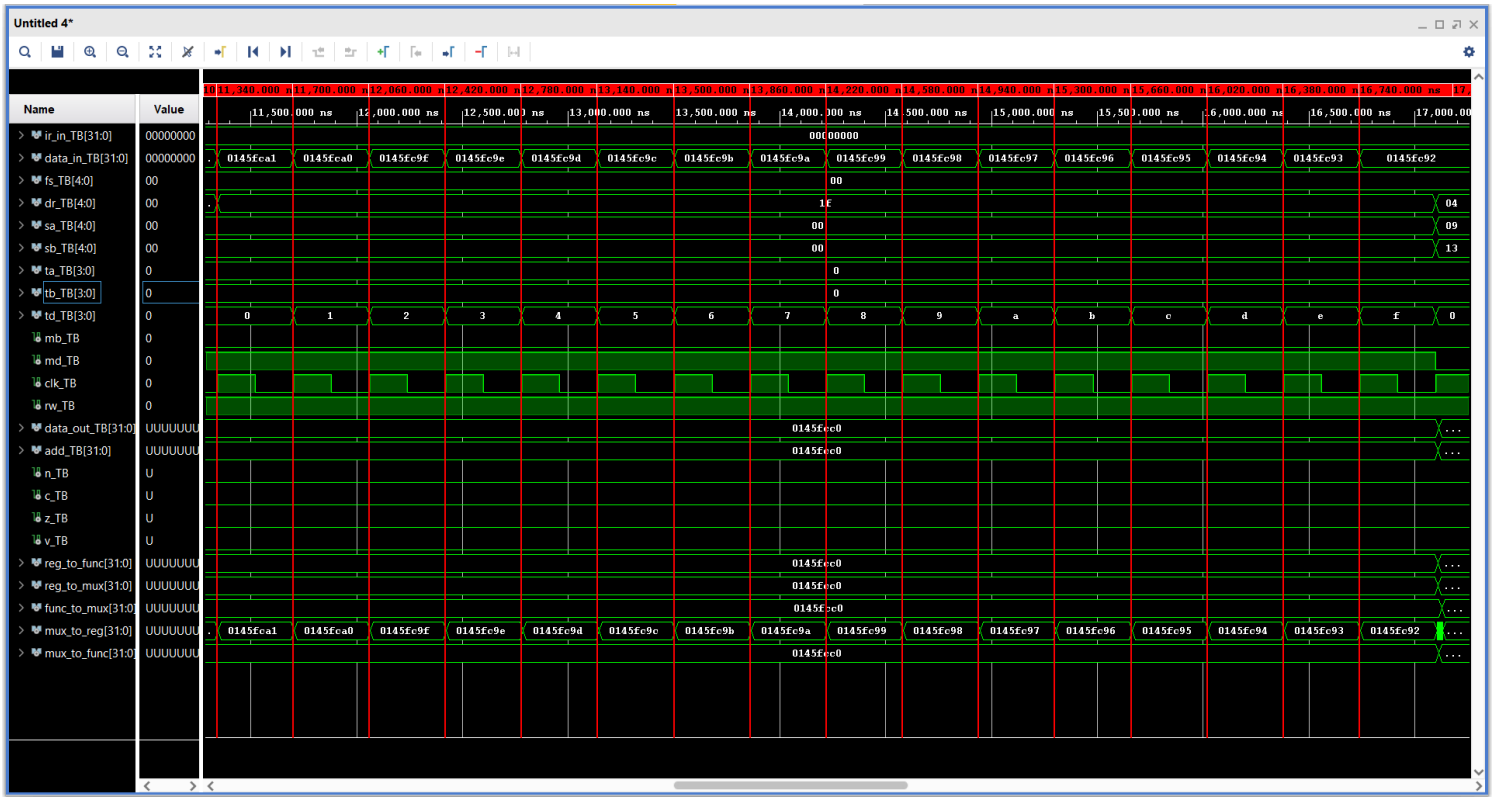
Each of the timing diagrams are shown below. For clarity, the first 15 operations are performed on $A = 0145FCB7$ and $B = 0145FCAD$, and the second 10 with $B = 0145FCC0$. The outputs of these operations are shown on signals $func\_to\_mux$ and $mux\_to\_reg$.

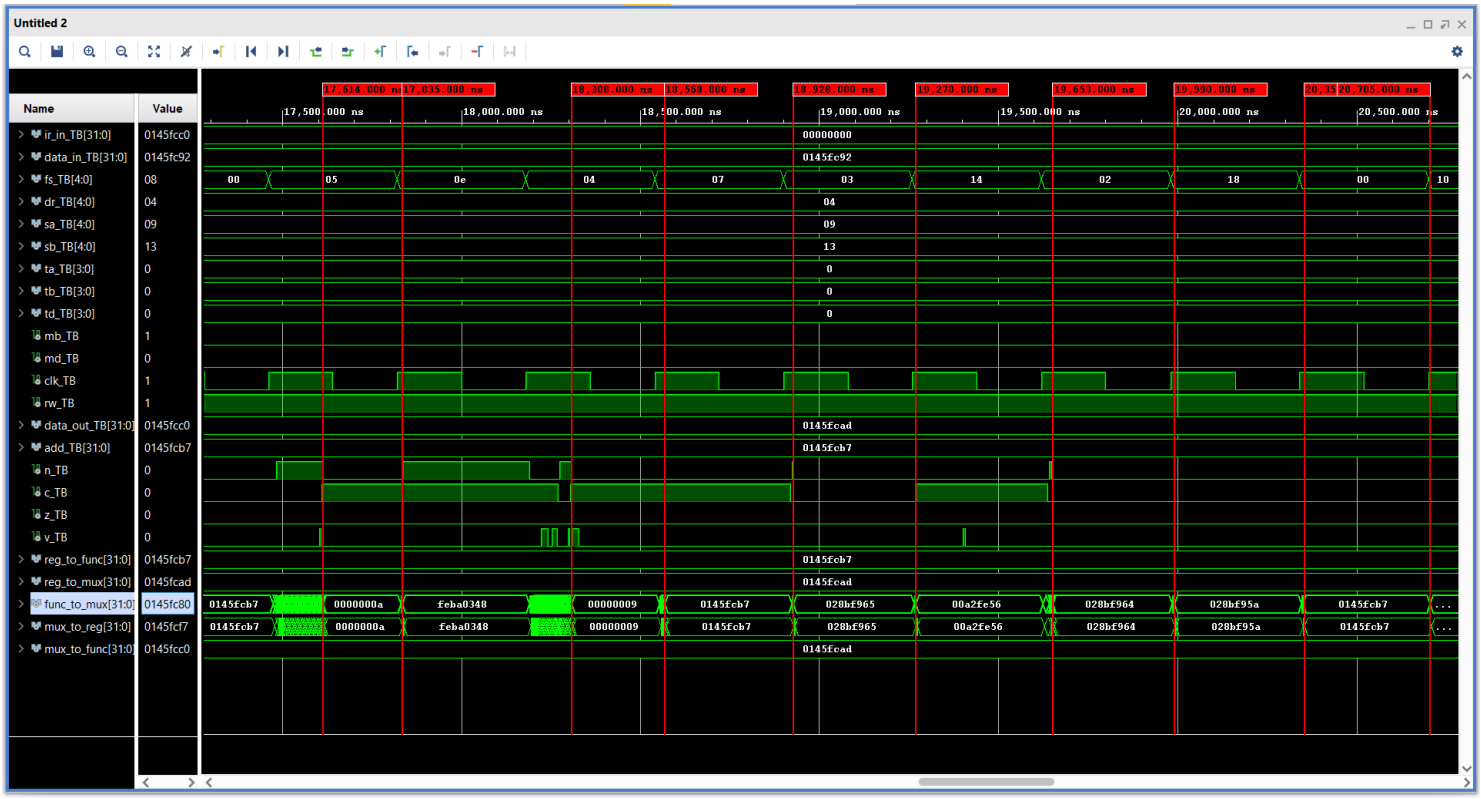
# DP_Datapath_21363904



# DP_Datapath_21363904



# DP_Datapath_21363904