COL216 Lab Assignment-2 (S4)

- Mayank Mangla (2020CS50430)

The aim of this project is to design hardware for implementing a processor that can execute a subset of ARM instructions described below. Starting with a skeleton design, the hardware is built in several stages, adding some functionality at every stage. The designs are to be expressed in VHDL and then simulated and synthesized.

Stage 4: In previous stage, we constructed a multi cycle processor.The previous stage program has been improved and improvised. Many programs have been tested on this new program ensuring to cover maximum of the cases.
The report for the stage 4 is given below:

1 Program Information

This program has been test on "eda playground" using "-2019 -o" flags.
This program contains the same number of files with the same description of the files in previous stages.

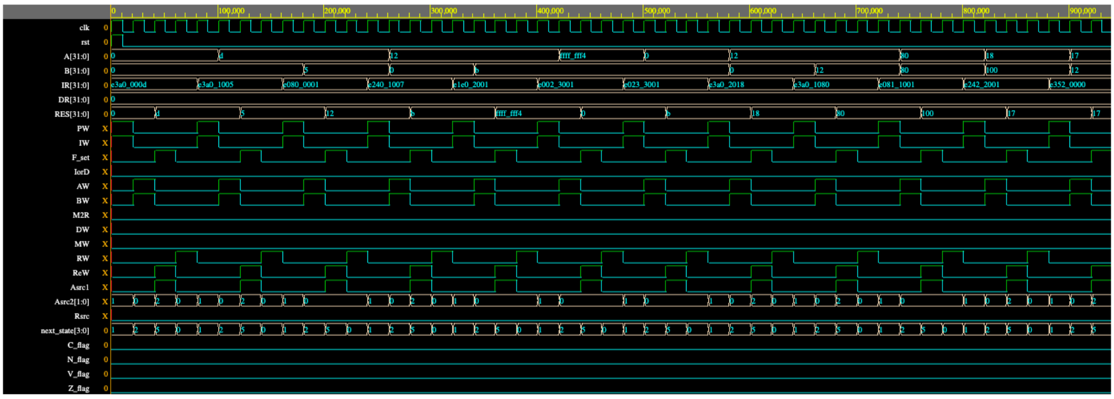
2 Test Cases

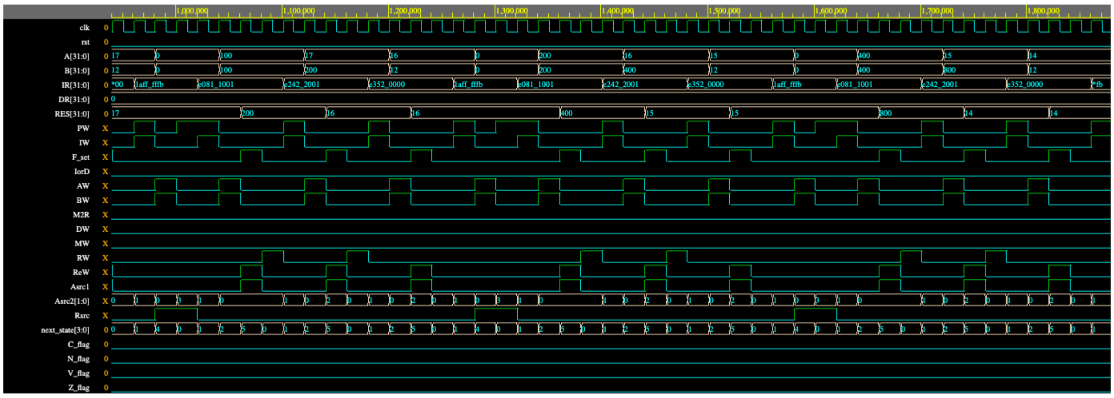
1. Case 1 This case was to check the setting of overflow flag and some of the basic dp operations. The binary code for the program is as follow

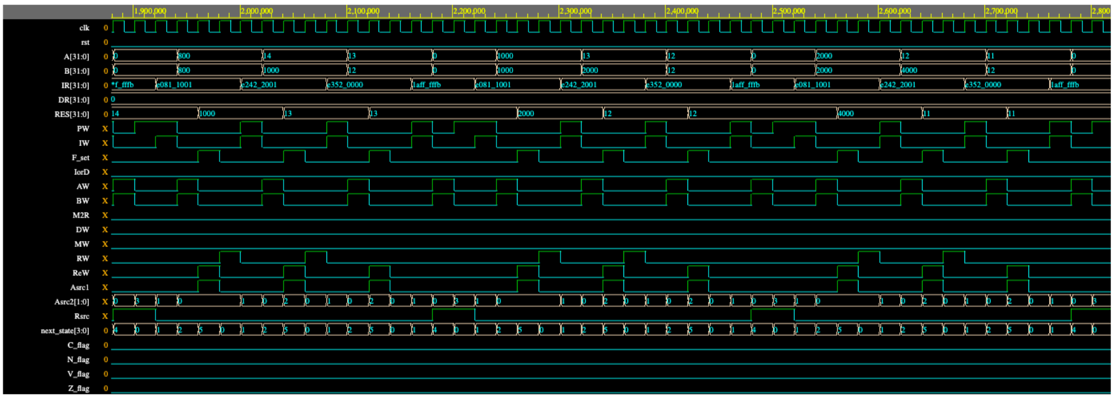
```
0 => x"E3A0000D",
1 => x"E3A01005",
2 => x"E0800001",
3 => x"E2401007",
4 => x"E1E02001",
5 => x"E0023001",
6 => x"E0233001",
7 => x"E3A02018",
8 => x"E3A01080",
9 => x"E0811001",
10 => x"E2422001",
11 => x"E3520000",
12 => x"1AFFFFFB",
13 => x"E0911001",
others => x"00000000"
```

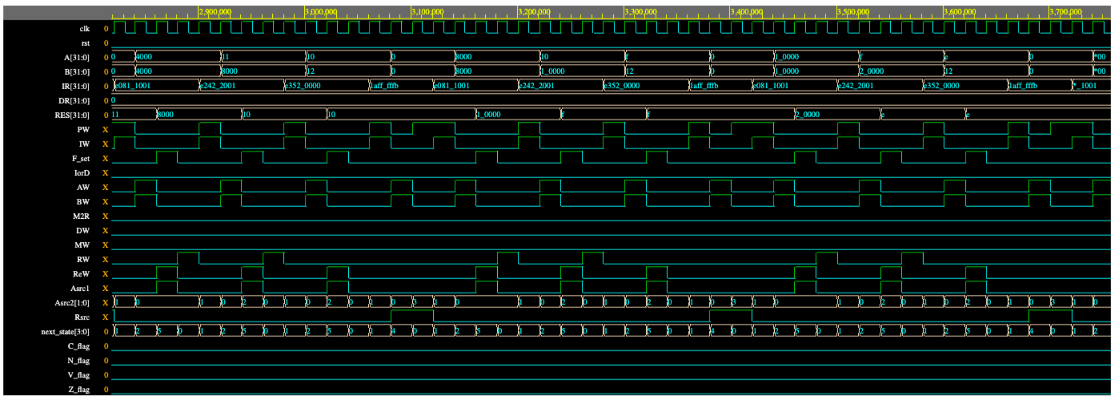
```
mov r0, #0x0d
mov r1, #5
add r0, r0, r1
sub r1, r0, #7
mvn r2, r1
and r3, r2, r1
eor r3, r3, r1
mov r2, #0x18
mov r1, #0x80
loop: add r1, r1, r1
sub r2, r2, #1
cmp r2, #0
bne loop
adds r1, r1, r1
```

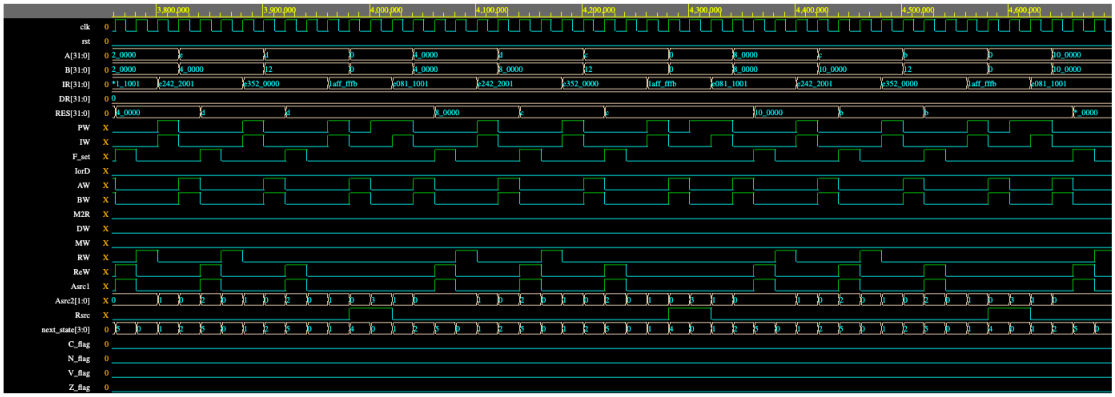
Below are the images of the ep wave:

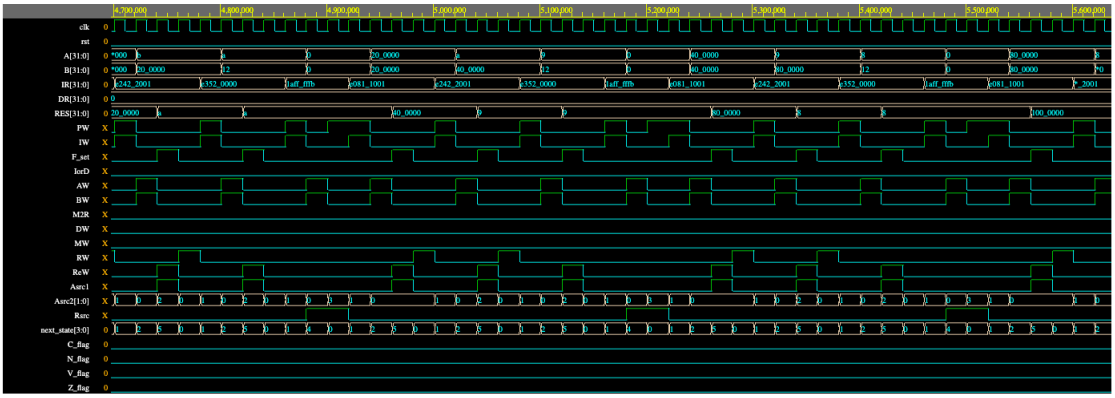


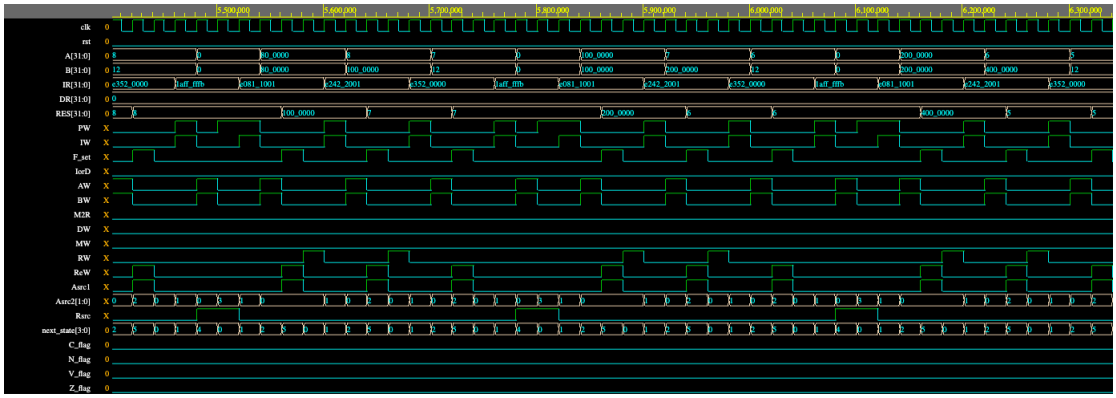


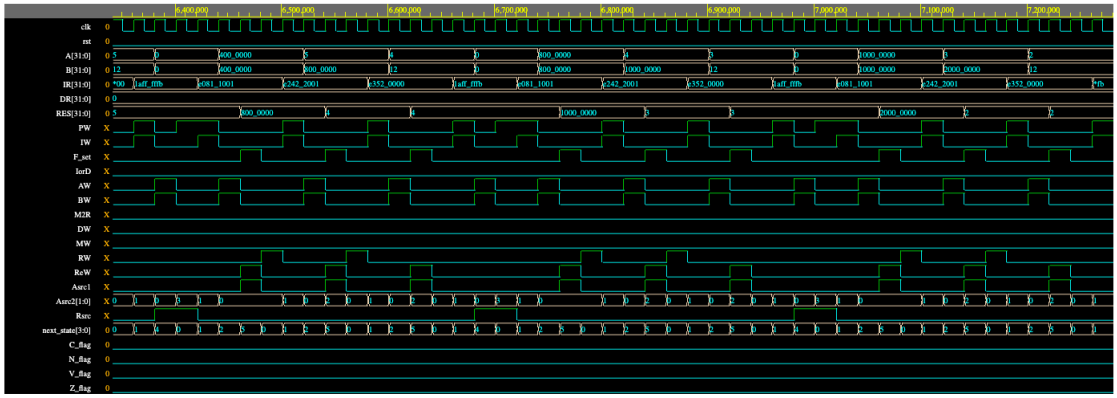


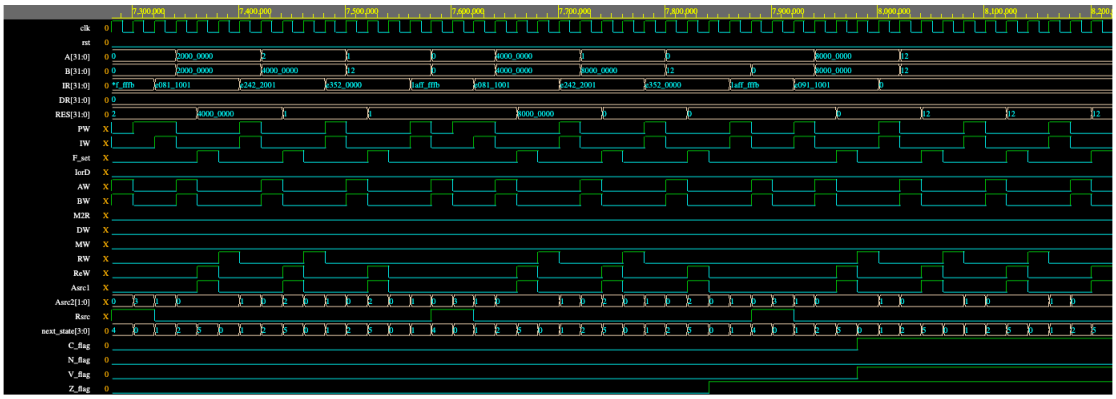



clk
rst
A[31:0]
B[31:0]
IR[31:0]
DR[31:0]
RES[31:0]
PW
IW
F_set
IorD
AW
BW
M2R
DW
MW
RW
ReW
Asrc1
Asrc2[1:0]
Rsrc
next_state[3:0]
C_flag
N_flag
V_flag
Z_flag


clk
rst
A[31:0]
B[31:0]
IR[31:0]
DR[31:0]
RES[31:0]
PW
IW
F_set
IorD
AW
BW
M2R
DW
MW
RW
ReW
Asrc1
Asrc2[1:0]
Rsrc
next_state[3:0]
C_flag
N_flag
V_flag
Z_flag


clk
rst
A[31:0]
B[31:0]
IR[31:0]
DR[31:0]
RES[31:0]
PW
IW
F_set
IorD
AW
BW
M2R
DW
MW
RW
ReW
Asrc1
Asrc2[1:0]
Rsrc
next_state[3:0]
C_flag
N_flag
V_flag
Z_flag


clk
rst
A[31:0]
B[31:0]
IR[31:0]
DR[31:0]
RES[31:0]
PW
IW
F_set
IorD
AW
BW
M2R
DW
MW
RW
ReW
Asrc1
Asrc2[1:0]
Rsrc
next_state[3:0]
C_flag
N_flag
V_flag
Z_flag

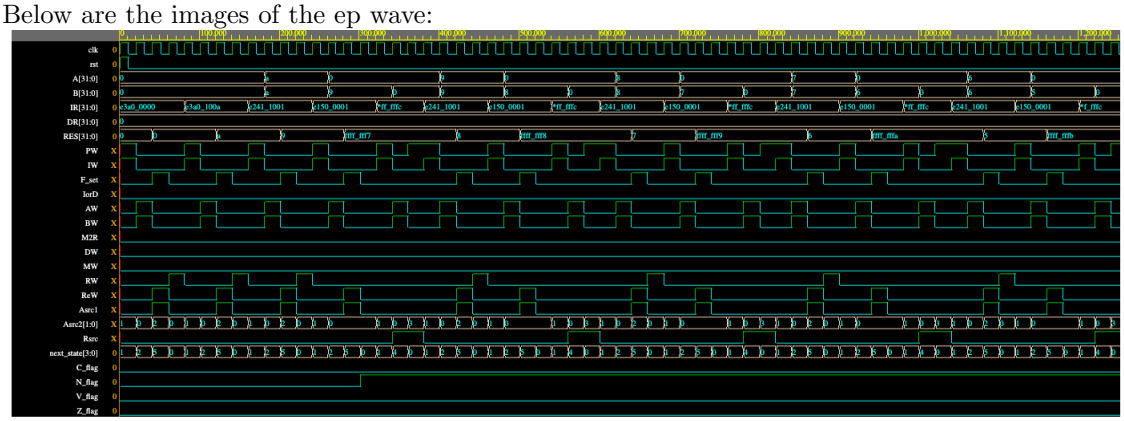


2. Case 2 Checking the branch instruction.

```
0 => x"E3A00000",
1 => x"E3A0100A",
2 => x"E2411001",
3 => x"E1500001",
4 => x"BAFFFFFC",
5 => x"E3A01000",
6 => x"E3A00000",
others => x"00000000"
```

```
mov r0, #0
mov r1, #10
loop: sub r1, r1, #1
cmp r0, r1
blt loop
mov r1,#0
mov r0, #0
```

Below are the images of the ep wave:

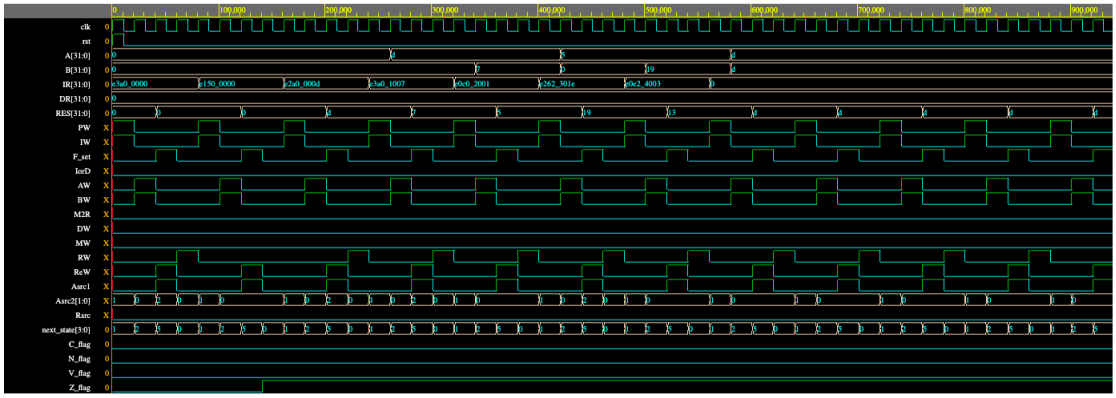


3. Case 3

```
0 => x"E3A00000",
1 => x"E1500000",
2 => x"E2A0000D",
3 => x"E3A01007",
4 => x"E0C02001",
5 => x"E262301E",
6 => x"E0E24003",
others => x"00000000"
```

```
mov r0,#0
cmp r0,r0
adc r0,r0,#13
mov r1, #7
sbc r2,r0,r1
rsb r3, r2, #30
rsc r4,r2,r3
```

Below are the images of the ep wave:

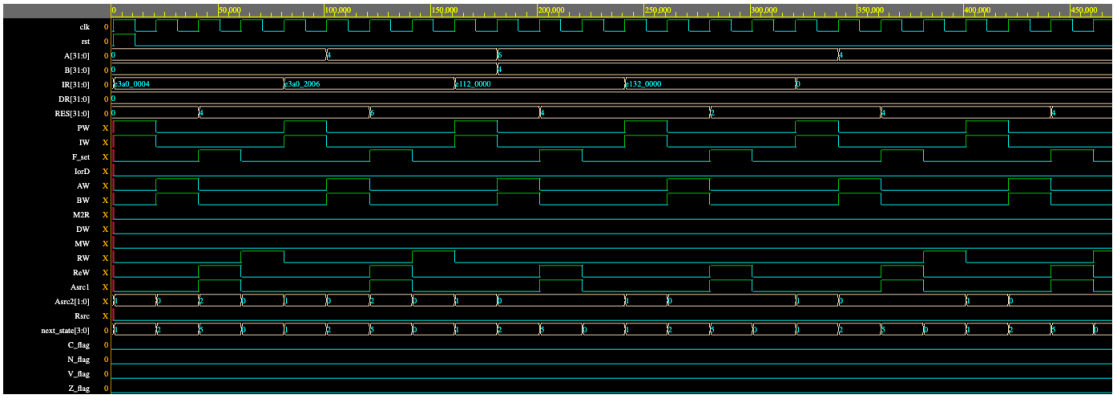


4. Case 4

```
0 => x"E3A00004",
1 => x"E3A02006",
2 => x"E1120000",
3 => x"E1320000",
others => x"00000000"
```

```
mov r0, #4
mov r2, #6
tst r2,r0
teq r2,r0
```

Below are the images of the ep wave:



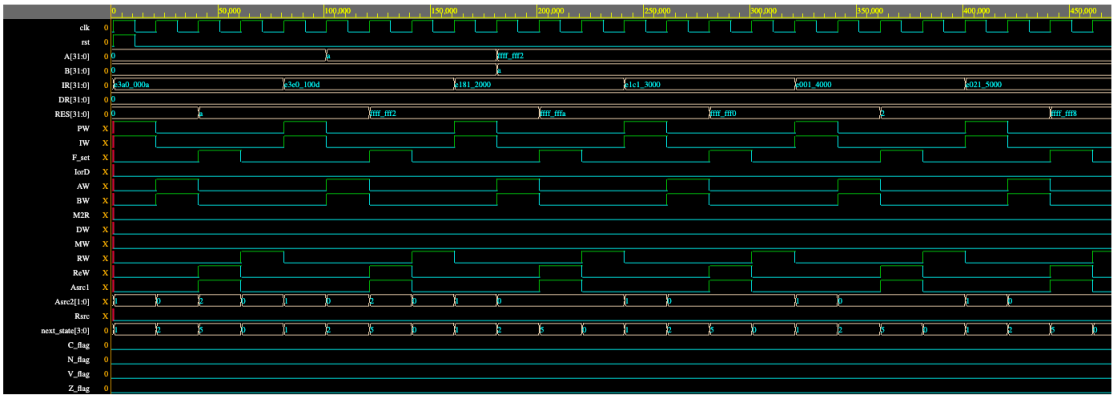
5. Case 5

```
0 => x"E3A0000A",
1 => x"E3E0100D",
2 => x"E1812000",
3 => x"E1C13000",
4 => x"E0014000",
5 => x"E0215000",
others => x"00000000"
```

```
mov r0,#10
mvn r1, #13
orr r2,r1,r0
bic r3,r1,r0
and r4,r1,r0
```

```
eor r5, r1,r0
```

Below are the images of the ep wave:

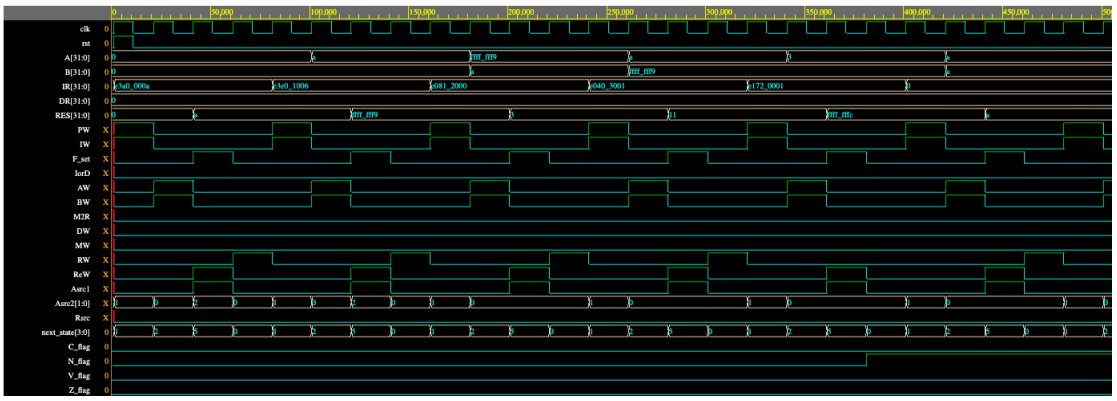


6. Case 6

```
0 => x"E3A0000A",
1 => x"E3E01006",
2 => x"E0812000",
3 => x"E0403001",
4 => x"E1720001",
others => x"00000000"
```

```
mov r0,#10
mvn r1, #6
add r2, r1,r0
sub r3,r0,r1
cmn r2,r1
```

Below are the images of the ep wave:



3 Conclusion

The program has given the correct result for the above test cases. The submission file (.zip) contains 12 files other than this report file.
Seven files each one for one component (e.g. alu_desgin.vhd), one contains the glue code (glue_code.vhd), one contains the test bench for the program (testbench.vhd), one contains the Finite state machine (fsm_design.vhd), one contains the data structures (given along with the assignment) (myTypes.vhd) and one is run.do file that has been used to synthesis.