COL216 Lab Assignment-2 (S5)

- 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 5: Until previous stage, we have constructed a multi cycle processor for some basic instruction set. We have now introduced some more instructions to our processor to execute. Shift/Rotate is introduced. For this we have created a new entity that shifts or rotates the data as required or returns the data as it is if no shift is required. A new register is defined that stores the shifted data, two new states in fsm are introduced that manage the cycle for different instruction. New register is defined to store the value of register in register file that states the shift amount in some cases.
The report for the stage 5 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. There were some changes in some of the entities and glue logic previously defined.

1. **FSM**:= Two states were introduced after the state where we distinguish between the paths of branch, DP or DT instruction that are common for DP and DT instructions.
   One state reads the shift amount stored in register specified in instruction bits. Other state gives the shifted or rotated output that is the required offset for the instruction.

2. **Reversal**:= a new entity definition is done. This entity reverses the 32 bit data. This entity is required in case of left rotation. This is used before and after the ROR in shift rotate unit.

3. **Shift Rotate Unit**:= a new entity definition is done. This entity shifts or rotates the data input with required amount and with given type of rotation/shift (LSL, LSR, ASR, ROR). This entity is made up of 5 entities which behaves similarly but with fixed amount of shift i.e., amount is 1, 2, 4, 8 and 16 for the 5 different units. Those 5 units are combined in this SRU which are enabled using the bits of shift amount.

The logs for the new entity "sru" (along with the required components) is given below

```
# Info: ********************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ********************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ------------------------------------------------------------
# Info: IOs                               76      210      36.19%
# Info: Global Buffers                    1       32        3.12%
# Info: LUTs                              189     63400     0.30%
# Info: CLB Slices                        39      15850     0.25%
# Info: Dffs or Latches                   33      126800    0.03%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ------------------------------------------------------------
# Info: ********************************************************************
# Info: Library: work    Cell: ShiftRotateUnit    View: implement_sru
# Info: ********************************************************************
# Info:  Number of ports :                          76
# Info:  Number of nets :                           355
# Info:  Number of instances :                      312
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                 33
```

```
# Info:  Number of LUTs :                              189
# Info:  Number of Primitive LUTs :                    201
# Info:  Number of LUTs with LUTNM/HLUTNM :             24
# Info:  Number of accumulated instances :             312
# Info: ************************************************************************
```

2 Test Cases

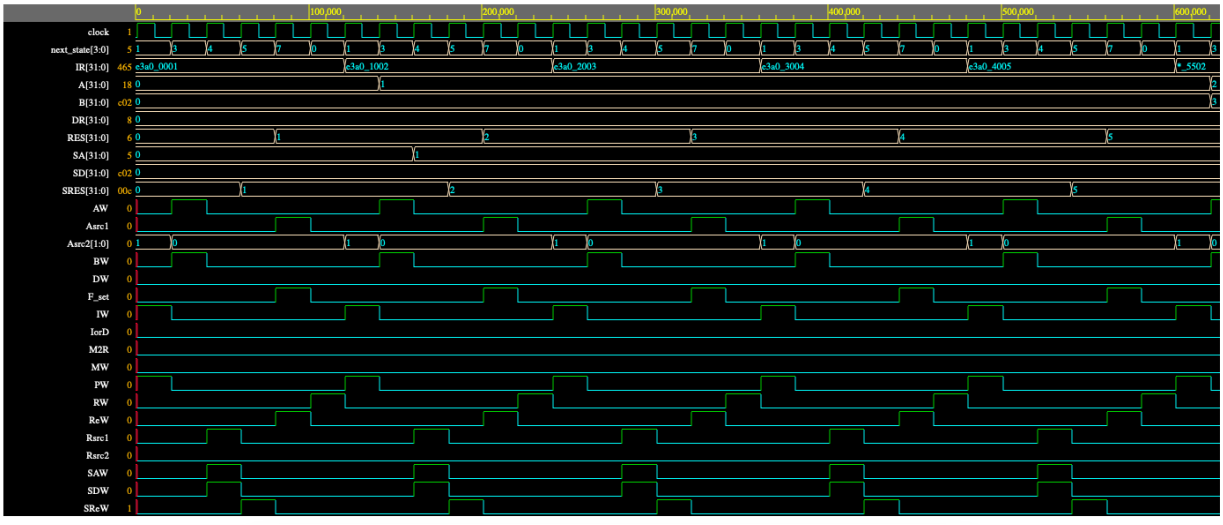
The program has been tested for this test case. This includes all the four types of shift instructions.
Instructions in Hex code

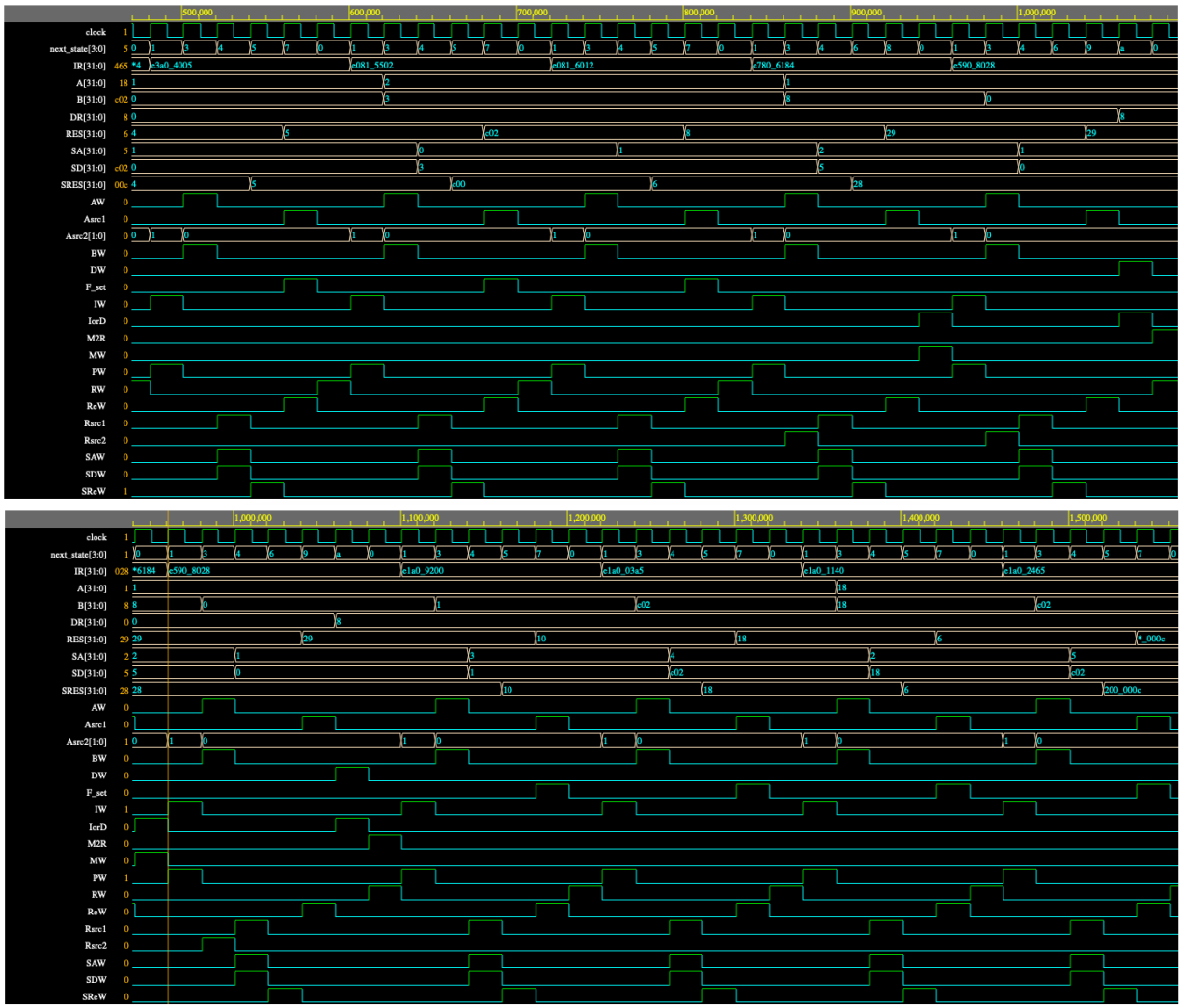
```
0 => x"E3A00001",
1 => x"E3A01002",
2 => x"E3A02003",
3 => x"E3A03004",
4 => x"E3A04005",
5 => x"E0815502",
6 => x"E0816012",
7 => x"E7806184",
8 => x"E5908028",
9 => x"E1A09200",
10 => x"E1A003A5",
11 => x"E1A01140",
12 => x"E1A02465",
others => x"00000000"
```

Instructions in ARM code

```
mov R0,#1
mov R1,#2
mov R2,#3
mov R3,#4
mov R4, #5
add R5, R1, R2, LSL #10
add R6, R1, R2, LSL R0
str R6, [R0, R4, LSL #3]
ldr R8, [R0, #40]
mov R9, R0, LSL #4
mov R0, R5, LSR #7
mov R1, R0, ASR #2
mov R2, R5, ROR #8
```

Below are the images of the ep wave for the above test cases:





3 Conclusion

The program has given the correct result for the above test cases. The submission file (.zip) contains 18 files other than this report file.
Eight files each one for one component (e.g. alu_desgin.vhd), five shift units are there which are component of the main Shift Rotate Unit, 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.