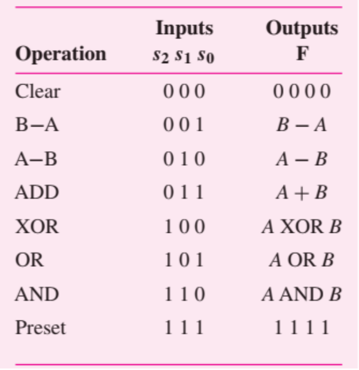
Assignment -06 : Procedural Assignments

Using Casex and Casez

- Write code for HEX to seven segment decoder using **case statement** .(Page 219, brown)
- Write code for ALU using **case statement**, refer following table



Operation	Inputs $s_2\ s_1\ s_0$	Outputs F
Clear	0 0 0	0 0 0 0
B−A	0 0 1	$B - A$
A−B	0 1 0	$A - B$
ADD	0 1 1	$A + B$
XOR	1 0 0	*A* XOR *B*
OR	1 0 1	*A* OR *B*
AND	1 1 0	*A* AND *B*
Preset	1 1 1	1 1 1 1

- **Simple Pattern Detector (casex) :** Design a module that uses a casex statement to detect a specific pattern (e.g., if the MSB of a 3-bit input is 1) and assert an output signal.
- **Instruction Decoder (casex) :** Create an instruction decoder that interprets a 4-bit opcode with don’t care bits using casex to generate appropriate control signals.
- **ALU Operation Selector (casex) :** Implement a basic ALU selector module where a 4-bit opcode chooses between arithmetic or logic operations using casex to handle uncertain input bits.
- **BCD to 7-Segment Display Decoder (casez) :** Build a decoder that converts a 4-bit BCD input into a 7-segment display output, using casez to manage any high-impedance or don’t care conditions in the input.
- **Digital Lock System (casez) :** Design a digital lock module that validates an input code using casez to allow for partial matching (don’t cares) in the sequence, outputting an unlock signal when the correct pattern is detected.
- **Priority Encoder (casez) :** Design a priority encoder that outputs the highest priority active input from an 8-bit signal, using casez to simplify handling of don’t care conditions.