1. Fazer o diagrama de uma máquina de estados para contador com uma entrada de A de 2 bits. A saida deve ser out=0,1,2,0,1,2 quando A=0, out=3,2,4,3,2,4 quando A=1, out=1,3,2,3,1,3,2,3 quando A=2 e out=5,6,3,5,6,3, ...quando A=3. Desenhar o diagrama com http://asciiflow.com/ ou outro editor ASCII e colocar no seu código Verilog com CASE. Escrever um testbench para testar usando seu numero de matricula. Se for 82322, converter para binario 1010000110010010, depois usar para alimentar A de 2 em 2, A=1,1,0,0,1,2,1,0,2. Proietar com memória e implementar em Verilog. Projetar com portas lógicas também. O testbench deve executar nas três implementações com mesmo resultado. Usar aplicativo de mapa de karnaugh ou executável como https://pyeda.readthedocs.io/en/latest/2llm.html para gerar as equações minimizadas.

minimizadas.													
	Estado				A 1	A 0	Proximo	s2	s1	s0	p2	p1	p0
0													
1													
2													
3													
4													
5													
6													
7													
8													
9													
10													
11													
12													
13													
14													
15													
16													
17													
18													
19													
20													
21													
22													
23													
24													
25													
26													
27													
28													
29													
30													
31													

2) Suponha uma entrada A de 1 bit e uma saída S de 3 bits. Se A=0, a saída gera o ciclo $0,3,2,4 \rightarrow 0,3,2,4 \dots$ Se A=1, a saída gera o ciclo $4,3,5,2 \rightarrow 4,3,5,2,\dots$

Faca a tabela

Diagrama de Estados

A	Estado	Proximo	Saída			

Usar a seguinte codificação em função da sua matricula. Primeiro converter em Octal sua matricula = 82322 decimal = 240622 octal. Suponha que sua máquina tenha os estados 0,2,3,4 e 5. Então o código de estado 0 será 2, o codigo do estado 2 será 4, o codigo do estado 3 será 0, o código do estado 4 será 6 e como o código do estado 5 não pode ser 2 (próximo na sequencia de matricula), incrementar para 3. Voce deve entregar a três implementações no mesmo código, com estados e case, com memória e com portas lógicas. Medir quantos operadores AND, OR, NOT terão as equações para próximo estado e saídas. Por exemplo, $S1 = a \& b \mid ! c$. Esta equação tem 3 operadores. $S2 = a \& b \& ! c \mid b \& !a$, terá 6 operadores, S1 = S2 seriam 9 operadores. Medir o total gasto para todas as equações.