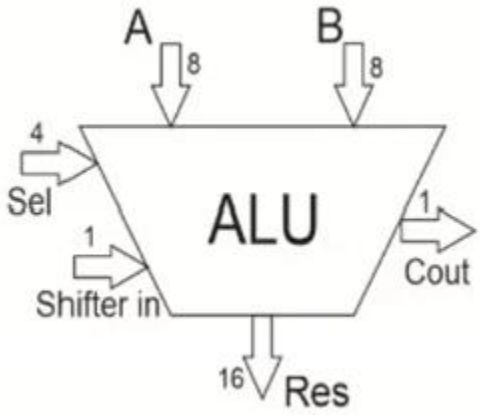
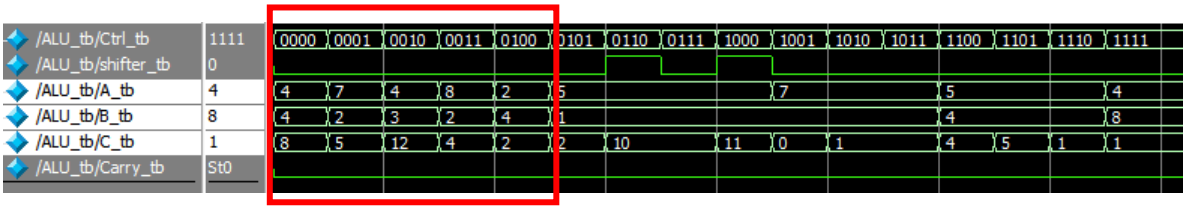
ALU



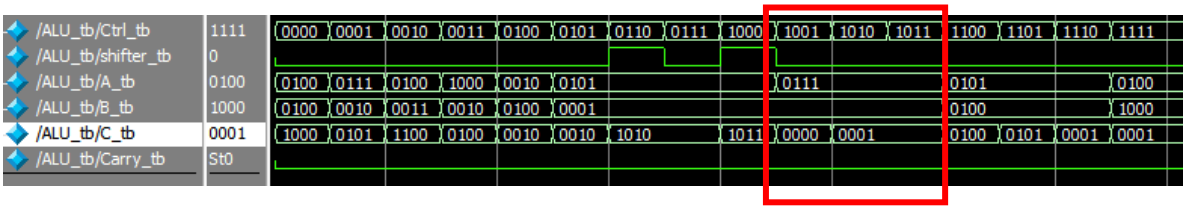
Selector	Operación
0000	A+B
0001	A-B
0010	A*B
0011	A/B
0100	A%B
0101	A>>B
0110	A>>B, Acarreo
0111	A<<B
1000	A<<B, Acarreo
1001	&A
1010	|A
1011	^A
1100	A&B
1101	A|B
1110	A^B
1111	00: A>B, 01: A<B, 10: A=B,

Se realiza la ALU de el mismo tamaño de la entrada, en clase se comentó que había esta posibilidad. A continuación, se muestra el código, así como el buen funcionamiento de la implementación.

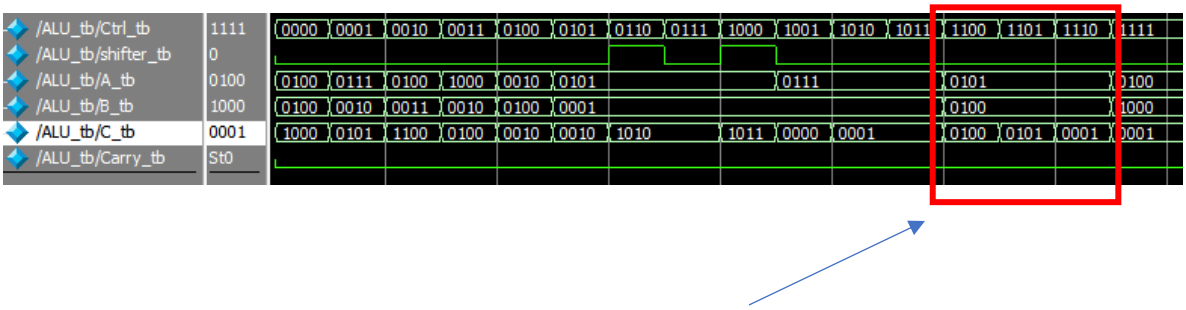



```
4'b0000: c_temp_reg = A + B;
4'b0001: c_temp_reg = A - B;
4'b0010: c_temp_reg = A * B;
4'b0011: c_temp_reg = A / B;
4'b0100: c_temp_reg = A % B;
```

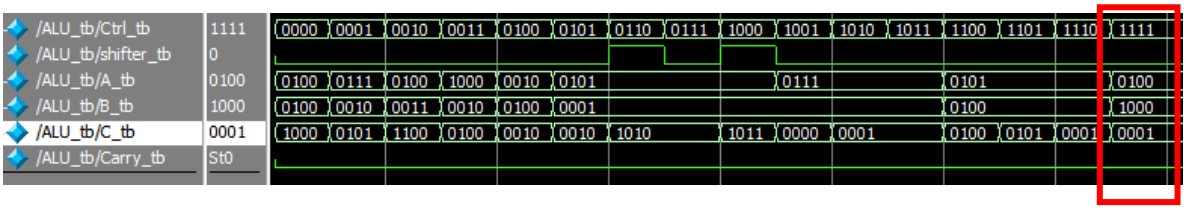

```
4'b0101: c_temp_reg = A>>B;
4'b0110:
if(shifter==1'b1)
    c_temp_reg = ~(~A>>B);
else
    c_temp_reg = A>>B;
4'b0111: c_temp_reg = A<<B;
4'b1000:
if(shifter==1'b1)
    c_temp_reg = ~(~A<<B);
else
    c_temp_reg = A<<B;
```




```
4'b1001: c_temp_reg = &A;
4'b1010: c_temp_reg = |A;
4'b1011: c_temp_reg = ^A;
```




```
4'b1100: c_temp_reg = A & B;
4'b1101: c_temp_reg = A | B;
4'b1110: c_temp_reg = A ^ B;
```




```
4'b1111: begin
   if(A>B)c_temp_reg = 0;
   else if (A<B)c_temp_reg = 1;
   else c_temp_reg = 2;
   end
default: c_temp_reg = 0;
endcase

if (A+B > mask)
begin
   carry_reg = 1'b1;
end
else begin
   carry_reg = 1'b0;
end
```

La practica tuvo un resultado satisfactorio, sin embargo, al momento de la realización se presentó un inconveniente que no se pudo resolver, el usar una entrada como variable. A pesar de este hecho, se utilizo una forma diferente de realizarla.