Carrera Maestría en Sistema Embebidos
Universidad de Buenos Aires
Procesamiento Digital de Señales. (PDS 2020)

Trabajo Práctico 1

Ejercicio 1 “Repaso VHDL” (2.5 pts):

Parte 1 ALU Simple:

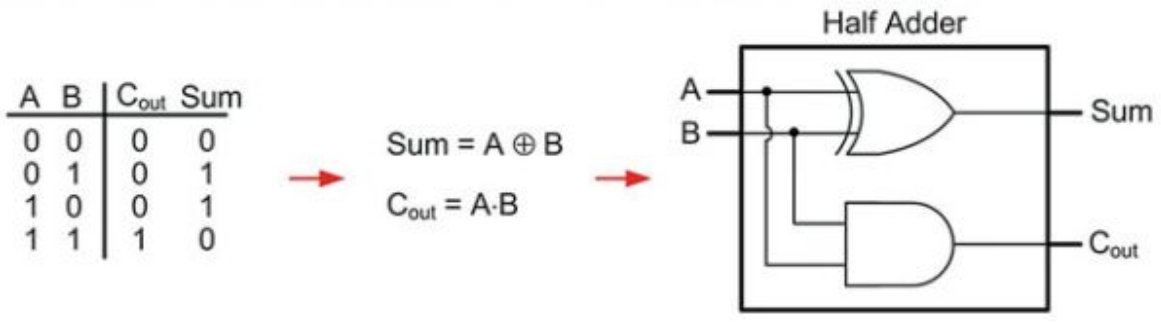
Diseñe una ALU teniendo en cuenta que las entradas son de 16 bits. La salida C también es de 16 bits.

SEL = “00” ⇒ C = A + B
SEL = “01” ⇒ C = A - B
SEL = “10” ⇒ C = A | B (bitwise OR)
SEL = “11” ⇒ C = A & B (bitwise AND)

- Entregar código RTL y código de verificación.

Parte 2 Half Adder:

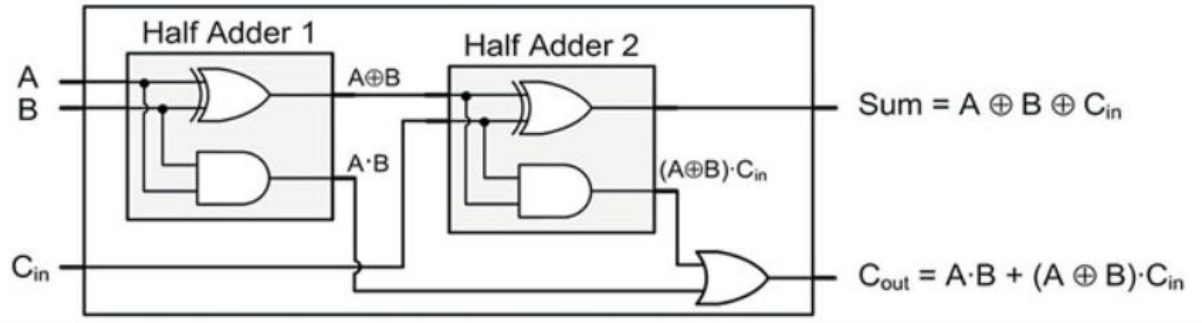
Escriba el código RTL de un Half Adder. El Half Adder calcula la suma teniendo en cuenta sólo 2 bits de entrada y nos da como resultado la suma y el carry out.



- Entregar código RTL y código de verificación verificando la tabla de verdad.

Parte 3 Full Adder:

Este circuito también genera una salida de suma y de carry out pero tiene como entrada 3 señales. (A,B y carry in). Con el siguiente circuito podemos realizar un full adder utilizando dos half adder.

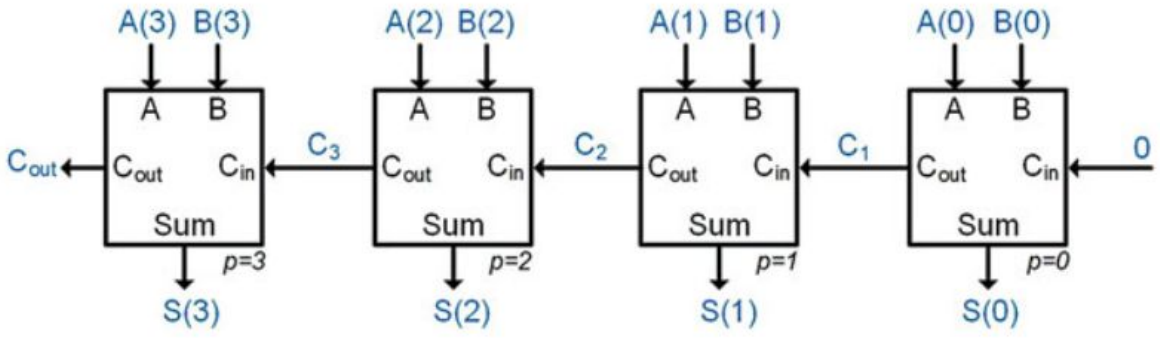


C_{in}	A	B	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

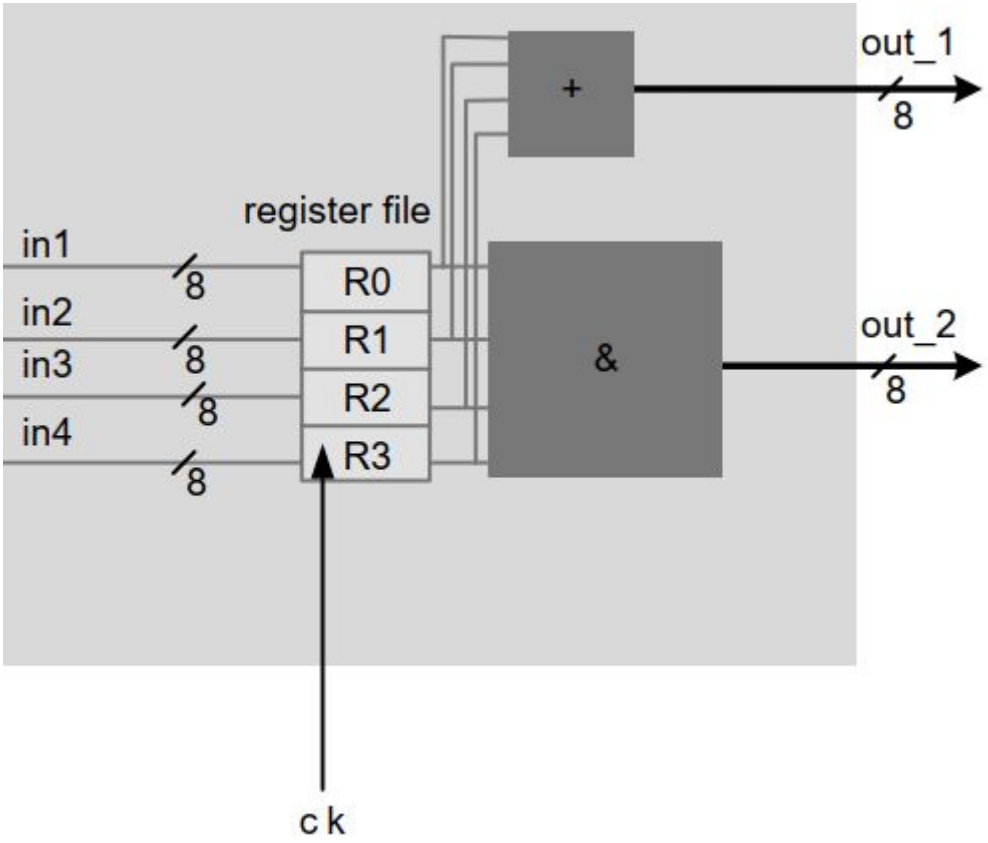
- Entregar código RTL y código de verificación verificando la tabla de verdad.

Parte 4 Ripple Carry Adder 4 bits:

Utilizando los bloques generados en las Parte 2 y 3 generar la siguiente descripción estructural



Parte 5 Sistemas sincrónicos



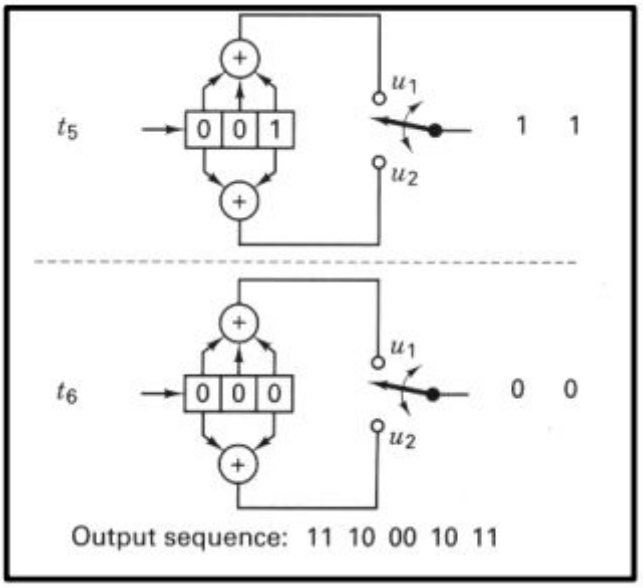
Las 4 entradas in1,in2,in3,in4 son entradas de 8 bits no signadas a los registros R0,R1,R2,R3. Cada salida es de 8 bits.
Tenga en cuenta que “&” es la and bit a bit.
Tenga en cuenta que “+” es la suma aritmética.
Los registros tienen reset asíncrono activo bajo.

- Entregar código RTL y código de verificación.

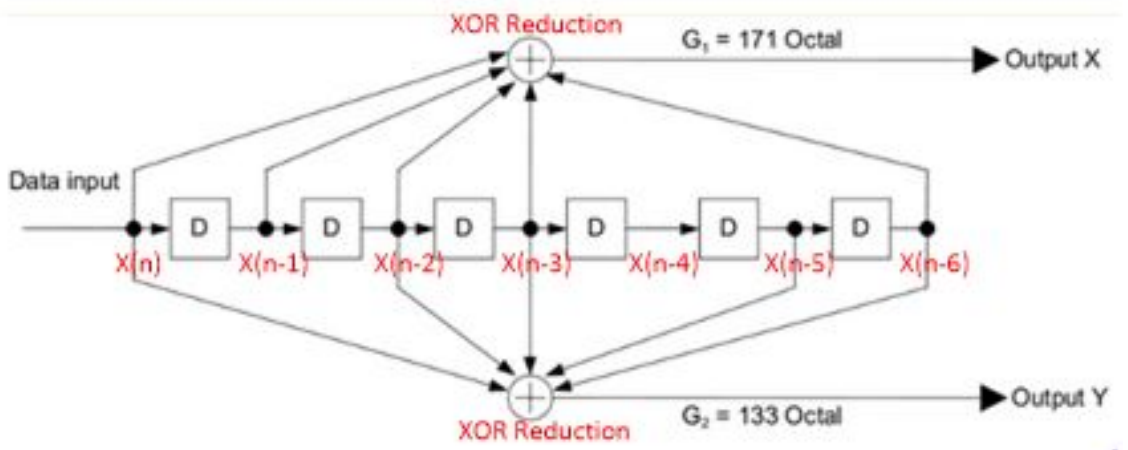
Ejercicio 2 “Encoder Convolucional” (2.5 pts)

Los códigos convolucionales son utilizados en telecomunicaciones para poder obtener redundancia a nivel de bits.
Lo bueno de estas implementaciones es que tienen una implementación sencilla. Se complejiza el decoder el cual no es el alcance de este ejercicio. (Por ejemplo viterbi).
Se entiende como los polinomios generadores como la suma la bitwise XOR entre las ramas.

Por ejemplo para el caso de abajo serían
P1 → 3b111
P2 → 3b101



Se solicita:
Realizar el código convolucional de la figura con los siguientes polinomios generadores.:



P1 = 171 OCTAL (Ver el gráfico de la figura). 0b0111_1001 en binario.
P2 = 133 OCTAL (Ver el gráfico de la figura). 0b0101_1011 en binario.
K = 7, orden del shift register.
R = ½. Input = 1 bits Output = 2 bits. O por cada bit de entrada salen 2 bits a la salida.
Entrada AXI Stream ancho de word 16 bits.
Salida AXI Stream ancho de word 32 bits.
Reset asincrónico activo bajo.

Hint 1: Para poder generar los vectores de pruebas para probar su implementación se pueden usar los siguientes módulos de Python / Matlab.
Commpy-Funciones --> trellis y conv_encode
scikit-dsp-comm --> fec_conv
Matlab

Hint 2: Como el código descrito tiene solamente un bit de entrada y ustedes están entrando con un word de 16 bit decidan cual es el bit que toman como el primero el menos significativo o el más significativo.

Hint 3: VHDL

Entregar:

- Realiza el código RTL
- Realizar el código de verificación (testbench)
- Código de generación de vectores de prueba con Python.
- Realizar el RTL Design en Vivado. Colocar en el reporte la captura.

Ejercicio 3 “AXI Stream y Ecuaciones de Diferencias” (2.5 pts)

Para la siguiente ecuación de diferencias:

$$y[n] = x[n] - x[n-1] + x[n-2] + x[n-3] + 0.5y[n-1] + 0.25y[n-2].$$

Análisis de sistemas lineales:

- Realizar la Transformada Z y obtener H(Z)
- ¿Puede decir que esta ecuación de diferencias representa un sistema FIR o IIR?. ¿Por qué?
- Realizar el diagrama de polos y ceros. ¿Qué conclusiones saca visualmente del diagrama de polos y ceros?.
- ¿Qué puede decir de la estabilidad del sistema?.
- Dibujar la respuesta en frecuencia, en magnitud y fase.

Implementación en Lógica Programable:

Teniendo en cuenta la siguientes especificaciones
La entrada al sistema es un bus AXI Stream de 8 bits. Entrada signada Q1,7.
La salida al sistema es un bus AXI Stream de 16 bits. Salida signada Q4,7.
Realice los multiplicadores utilizando “shift operations”. Por ejemplo para 0.5 hay que hacer un shift de 1 posición.
Que tenga reset asincrónico activo bajo.

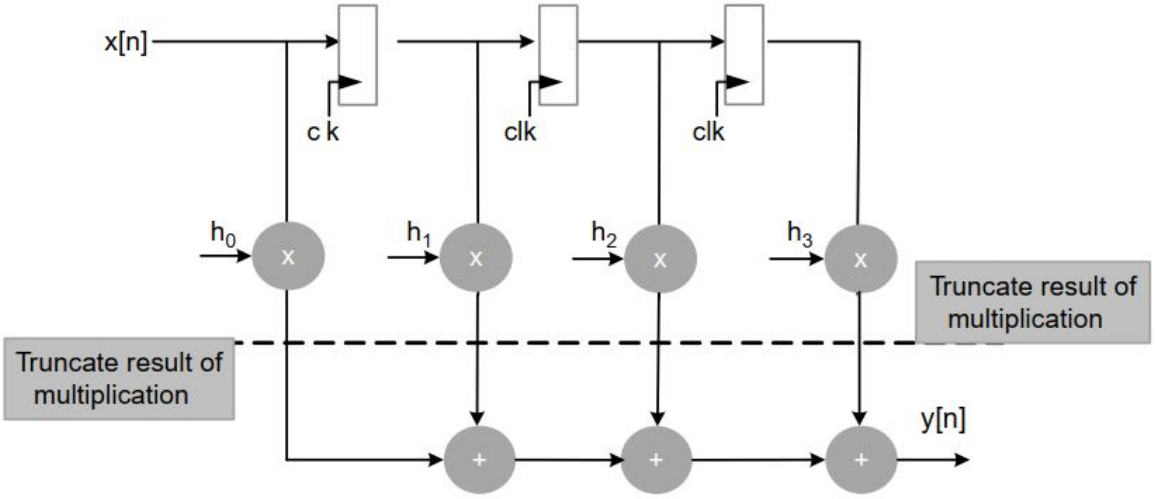
- Realice el código RTL que implemente la ecuación de diferencias.
- Realice el código testbench que le permita verificar el funcionamiento del filtro.
- Excitar el sistema con las siguientes señales:
    - DC.
    - Frecuencia Nyquist..
    - Frecuencia Nyquist / 2.
- Realizar el RTL Design en Vivado. Colocar en el reporte la captura.

Ejercicio 4 “FIR Simétrico AXI S” (2.5 pts)

Análisis de sistemas lineales:

Diseñar un filtro FIR de fase lineal siguiendo la siguiente plantilla:
Tipo de Filtro: Pasabajos
Método de diseño : Equiripple (Orden Mínimo)
Frecuencia de banda de paso: 0.2 fny
Frecuencia de banda de rechazo: 0.6 fny
Ripple en banda de paso: 1 db
Atenuación en banda de rechazo: 20 db
Utilice la herramienta de diseño de filtros que considere adecuada, FilterDesign (@Matlab) o pyfda (@Python)

- ¿Cuál es el orden del filtro obtenido?
- Dibujar la transferencia del filtro en magnitud y fase.
- Realizar la Transformada Z y obtener H(Z)
- Realizar el diagrama de polos y ceros. ¿Qué conclusiones saca visualmente del diagrama de polos y ceros?.



Implementación en Lógica Programable:

Teniendo en cuenta las siguientes especificaciones y el diagrama en bloques anterior.
Cuantizando los coeficientes teniendo en cuenta que son signados con una precisión Q1.15.
La entrada al sistema es un bus AXI Stream de 16 bits. Entrada signada Q1,15.
Seguir como arquitectura la indicada en la figura.
Realice el truncamiento en la multiplicación para que la salida de la multiplicación de los coeficientes con las muestras de entrada sea de 18 bits. Teniendo en cuenta esto define el formato para y[n]
Que tenga reset sincrónico activo alto.

- Realice el código RTL que implemente este filtro.
- Realice el código testbench que le permita verificar el funcionamiento del filtro.
- Realizar el RTL Design en Vivado. Colocar en el reporte la captura.
- Excite el sistema con una senoidal de 0.1 f_ny y 0.8 f_ny
- Repita los puntos 5,6,7 y 8 teniendo en cuenta la simetría del filtro.
- Repita los puntos 5,6,7 y 8 realizando redondeo en vez de truncado.