# Universidad de Costa Rica

Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0323 – Circuitos Digitales II
II ciclo 2021

Proyecto 1

PHY_PCI

Estudiante:
Meybelle Castro Valverde - B91887
Adrián Montero Bonilla - B85092
Sofía Villalta Jinesta - B88565
Sinaí Zamora Vega - B88719
Grupo 07

Profesor: Jorge Soto

23 de octubre, 2021

# Índice



# 1. Descripción arquitectónica

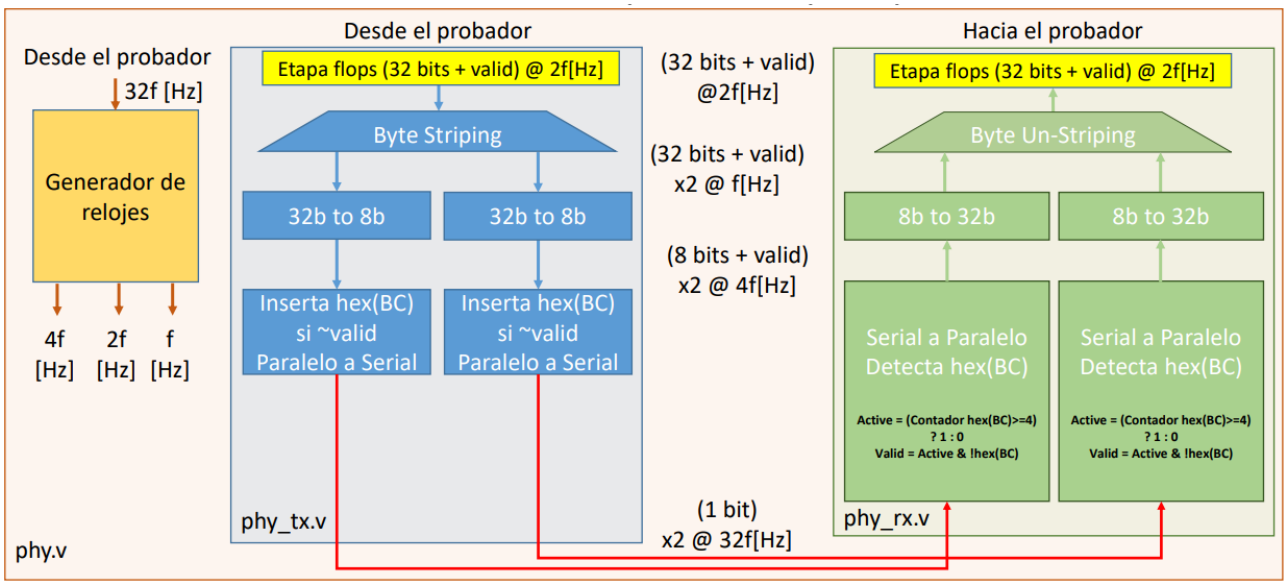


Figura 1: Esquemático del diseño.

# 2. Instrucciones de simulación

Primero, se debe descargar el .zip adjunto a este reporte y descomprimirlo en algún lugar. Una vez hecho esto, se abre dicha carpeta y dentro de esta se encuentran otras carpetas en las cuales se tienen guardados los módulos del proyecto. Para simular algún módulo, se abre

la terminal con la dirección de la carpeta deseada y se escribe ”make”; automáticamente se desplegará la ventanilla de GTKWave para verificar el módulo.

# 3. Resultados y análisis

## 3.1. Phytx

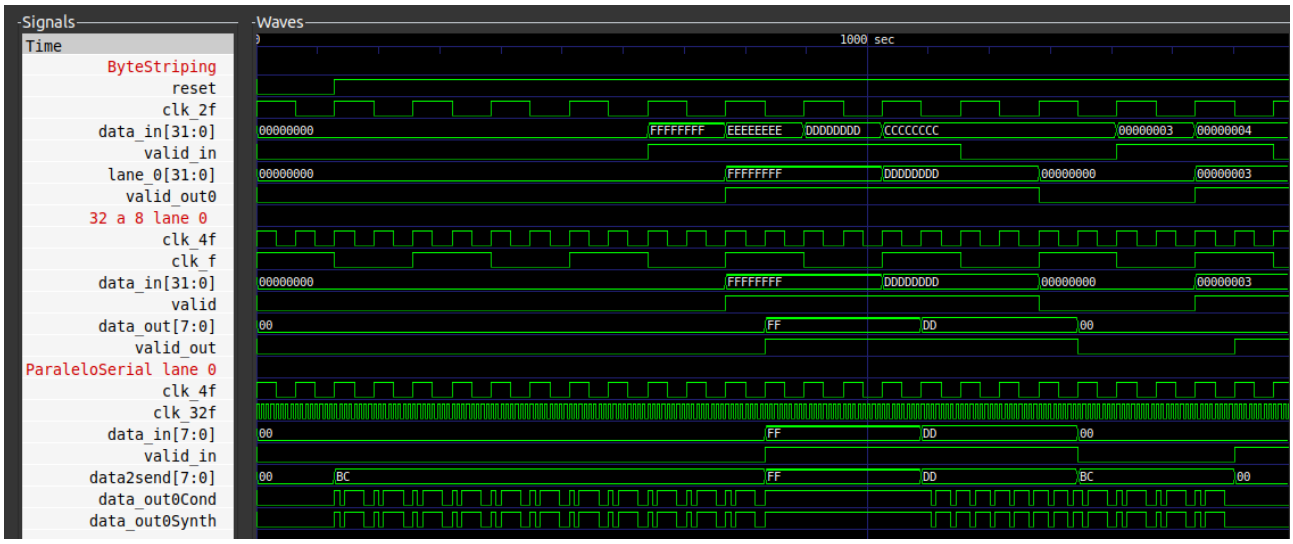


Figura 2: Resultado de Phytx.

Para esta parte lo que se hizo fue unir conductualmente los módulos, al principal llamado phytx se le asignan las entradas del byte striping y las salidas del paralelo serial, después se definieron los cables internos de cada módulo y se conectó cada parte instanciando con los nombres correctos del archivo. Las entradas de los módulos de 32b a 8b se conectaron a las salidas del byte striping y las entradas del paralelo serial se conectaron a las salidas de los módulos de 32b a 8b.

El comportamiento esperado del módulo phytx se ve muy bien en los resultados obtenidos, se puede observar como el data out conductual y el estructural funcionan exactamente igual, lo que quiere decir que el archivo fue bien sintetizado,también se puede observar que los datos se están transfiriendo de la forma esperada debido a que las entradas son válidas a partir de que el valid in está en alto durante 4 flancos del reloj, estas entradas son aceptadas y transmitidas a la salida de forma serial, como se evidencia en la imagen, también se puede observar que mientras el valid es cero la salida está sacando serialmente BC que es el comportamiento esperado.

### 3.1.1. Byte striping

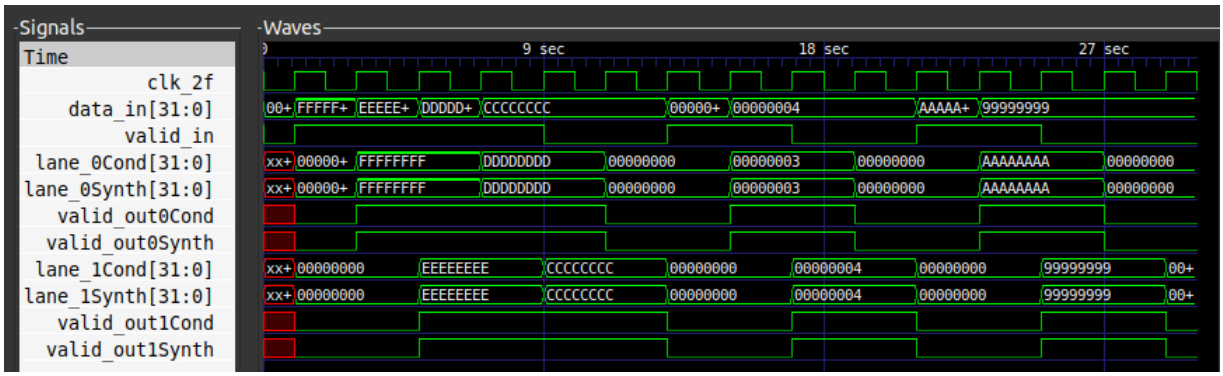


Figura 3: Diagrama de tiempos del byte striping

El byte striping recibe un bus de 32 bits como entrada y este lo que hace es que separa dicho bus en 2, a estos buses de salida se les llama lanes. Nótese que, mientras los datos de entrada sean válidos, este siempre saca el primer dato de data in en el lane 0 y luego pasa intercalando entre este mismo y el lane 1. Una peculiaridad de este diseño es que las salidas las saca un flanco de clk2f después, pero dicho delay no tiene ninguna repercusión alguna en la funcionalidad de dicho módulo.

Asimismo, tanto la descripción conductual como la estructural se comportan exactamente igual y, por consecuente, se puede afirmar que dicha implementación del byte striping es válida y sigue el comportamiento esperado.

### 3.1.2. 32b to 8b

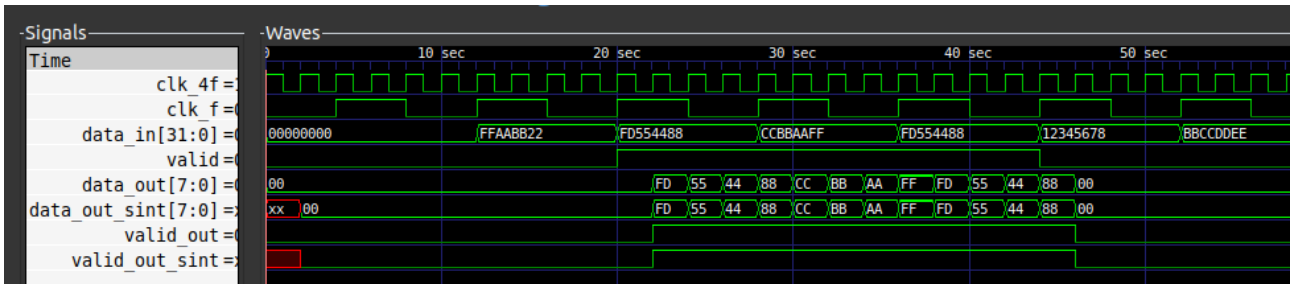


Figura 4: Diagrama de tiempos del 32b a 8b

Este modulo recibe la palabra de 32 bits y un valid. Así mientras el valid este activado la palabra recibida en esos momentos es valida y por lo tanto el modulo se encargara de sacar la señal en un bus de 8 datos, de esta manera tenemos una frecuencia menor con un bus de 8 datos. Esto sucede siempre y cuando el reset este activado, ya que cuando el reset = 0 la señal se convertirá en cero.

En la Figura 4 se puede observar el diagrama de tiempos del modulo 32b a 8b. Las pruebas realizadas para obtener este diagrama consisten en ingresarle palabras de 32b (la cual proviene de Byte Stripping), al segundo flanco creciente de el clk f se ingresa una palabra FFAABB22, pero el valid esta en cero, por lo que no se cuenta como valida para la salida. Luego en el tercer

flanco creciente del clk f se recibe la palabra FD55R4488 y en este mismo instante el valid se activa, por lo que, en la salida vemos reflejado esta palabra en 8 bits con un cierto retardo. Esto continua sucediendo hasta que el valid vuelve a ponerse en cero, por lo que aunque se reciban palabras, la salida sera cero.

### 3.1.3. Paralelo a serial

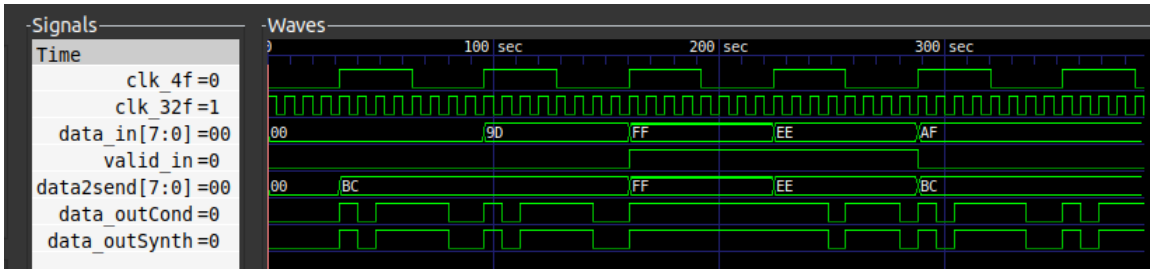


Figura 5: Diagrama de tiempos del paralelo a serial

Este módulo recibe como entrada un bus de 8 bits y, si dicha entrada es válida, este convierte el bus de entrada en datos seriales; en caso de que las entradas no sean válidas este da como salida la señal hex(BC) en modo serial. Se puede observar de la figura anterior que dicho módulo hace precisamente lo descrito anteriormente. Asimismo, tanto el módulo conductual como el estructural funcionan de la misma manera y gracias a esto se concluye que dicha descripción propuesta es correcta.

## 3.2. Phyrx


Figura 6: Resultados del Phyrx.

El módulo receptor toma una señal serial y entrega una salida de 32 bits. Cuando la señal data_in del paralelo serial llega a cuatro códigos BC, la señal active se pone en 1. Una vez que llegan los códigos esperados, el módulo comienza a paralelizar los datos para entregar la salida. En la figura 6 se puede observar que hay ocasiones en los que valid_out del phyrx está en bajo, esto se debe a que en el serial paralelo del lane 1 no se reciben 4 hex(BC) y esto provoca que el active del lane 1 siempre esté en bajo; las entradas recibidas en el serial paralelo del lane 1 son inválidas. Dichas entradas se escogieron así para verificar el comportamiento de los valids a lo largo de todo el phyrx y, como se puede ver en la figura anterior, la salida obtenida es justo la esperada.

### 3.2.1. Serial a paralelo

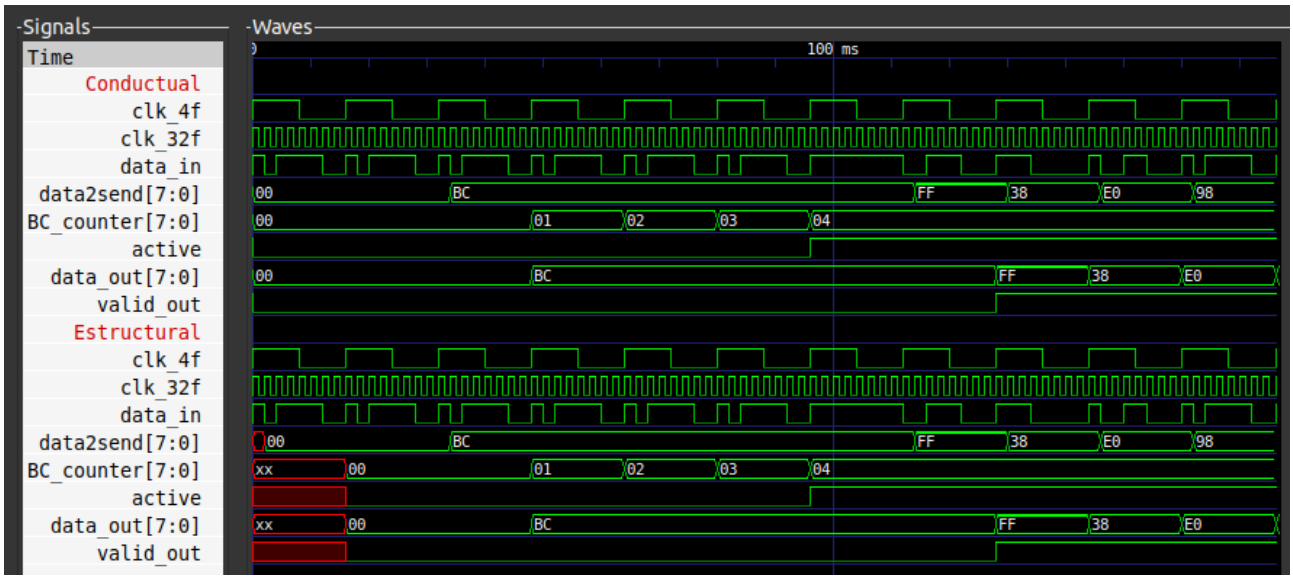


Figura 7: Diagrama de tiempos del serial a paralelo

El serial a paralelo lo que hace es que espera para recibir 4 señales de hex(BC) de forma serial y una vez que esto ocurre la señal de active se enciende. Con el active en alto, se empiezan a recibir las entradas de forma serial y solamente las que no sean hex(BC) son válidas; dichas entradas van a formar el bus de 8 bits de salida.

Este módulo fue muy problemático a la hora de diseñarlo debido a que no se comportaba como se quería en un principio. Para poder hacerlo funcionar de forma correcta, se le tuvo que insertar varios ciclos de delay antes de que entregue la salida, sin embargo, estos retardos no modifican el comportamiento esencial del módulo. Una vez más, el conductual como el estructural se comportan exactamente igual y se da como válida la solución de esta pieza.

### 3.2.2. 8b to 32b

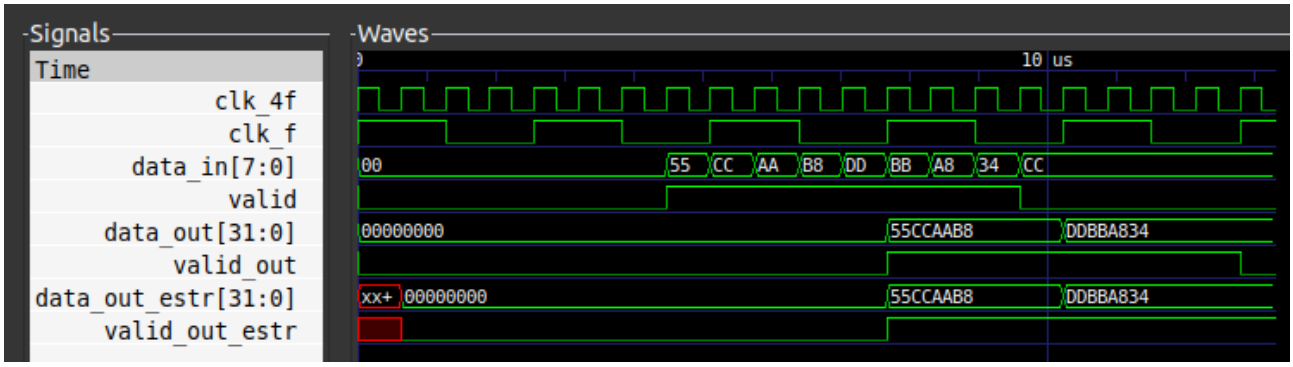


Figura 8: Diagrama de tiempos del 8b a 32b

Este módulo recibe de entrada una señal paralela a 4f y un valid. De este modo, siempre que valid esté arriba los datos del bus van a ser válidos, independientes del cuál sea el valor. En general, su comportamiento es inverso al 32b a 8b, ya que este va a reducir la frecuencia de la señal debido a que se está generando un aumento en el bus de datos. En su salida posee un bus de datos y un valid.

En la imagen 8 se observa la prueba que se hizo, esta consistió en agregarle una señal que proviene del serial paralelo con frecuencia de 4f; se inicia con un valid en 0, con lo cual se espera que la salida data_out y valid_out estén en 0. Una vez que valid esté en 1, se espera que se reflejen los datos de la entrada data_in en el data_out, pero, ahora conformados por 32 bits, en lugar de 8 bits. Y como parte de la comprobación se baja en ocasiones el valid de entrada y se espera que el valid de salida también lo haga.

### 3.2.3. Byte unstriping

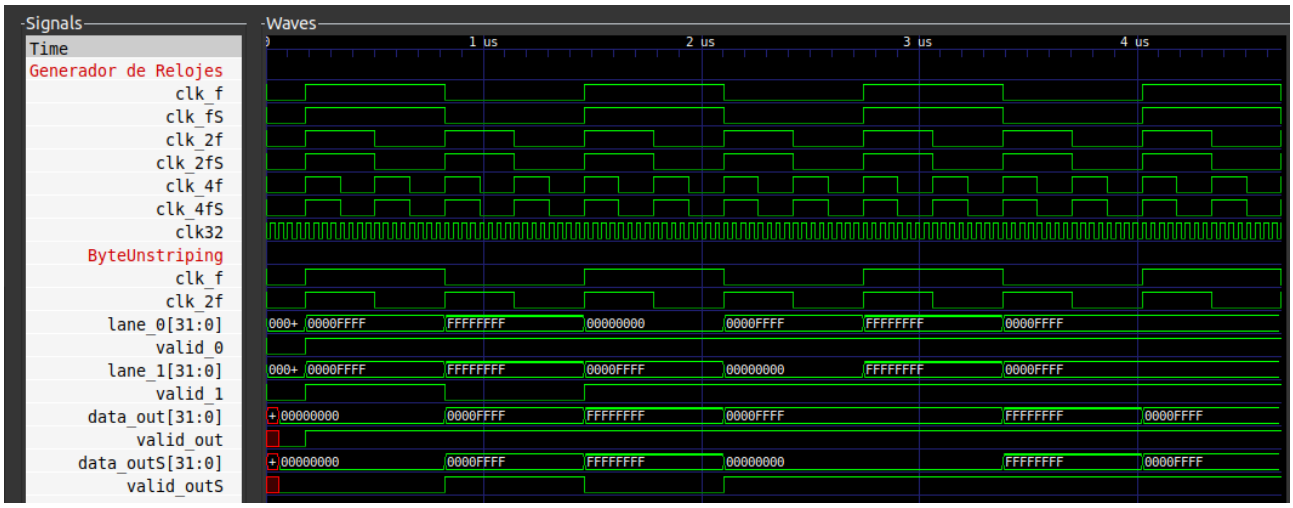


Figura 9: Diagrama de tiempos del byte unstriping y del generador de relojes

Lo que hace el byte unstriping es recompilar datos de ambas entradas (lanes) a una sola, lo cual se puede ver en la imagen, sin embargo, la salida estructural no da completamente bien, se puede observar que la salida conductual tiene una diferencia a la estructural en el cuarto flanco

de reloj; dicha discrepancia entre resultados se piensa que es por los delays y también por el funcionamiento de los valids. Esta diferencia entre salidas afectó más adelante al sintetizar el archivo phy, no obstante, nótese que el módulo conductual funciona correctamente.

De aquí se puede notar que no todas las descripciones conductuales van a dar igual a la hora de sintetizarlas y es por esto que se debe revisar dicho módulo más a detalle; por temas de tiempo, dicha descripción conductual se dejó tal y como se puede observar en la figura anterior.

### 3.3. Phy

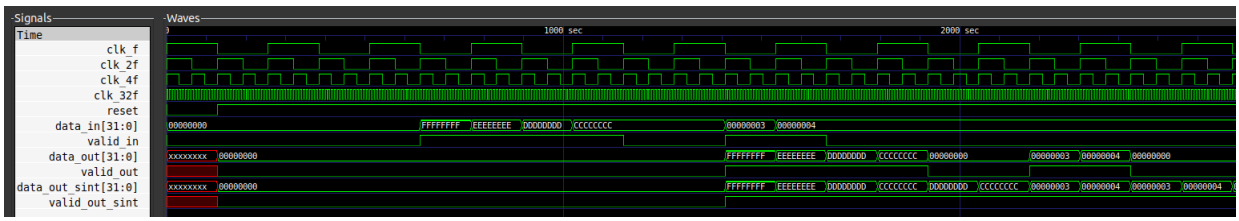


Figura 10: Resultados obtenidos para el módulo phy final.

La gran mayoría de los módulos creados para este proyecto cumplen con las especificaciones de diseño, sin embargo, el resultado conductual y estructural del phy grande difieren un poco. Se sospecha que dicho error pudo ser causado debido a la manera en la que se implementó la lógica de los valids en el byte unstriping, ya que a la salida de la descripción estructural se tiene que esta siempre es válida, cosa que no es verdad; el resultado correcto es el demostrado por la descripción conductual.

De igual forma, podemos ver que el el data_in y el valid_in se activan en el sexto flanco creciente del clk_2f, por lo que estas palabras ahora son validas y deben reflejarse en la salida, y como podemos observar en la Figura 10, la salida refleja la entrada con un retardo debido a todo el proceso por el que se debe someter antes de salir del phy_rx, luego, cuando el valid_in se pone en cero, la entrada ya no es valida y la salida se pone en cero.

## 4. Conclusiones

La respuesta conductual del phy grande no es igual a la estructural; dicho problema pudo haber surgido debido a un fallo con la lógica de los valids en la descripción estructural del byte unstriping. No obstante, el comportamiento general del phy grande sintetizado fue muy cercano al conductual (este refleja el comportamiento correcto); con un poco más de tiempo se podría corregir el error en dicho módulo para obtener ambas descripciones iguales.

Se recomienda para futuras entregas revisar más a fondo la implementación de la lógica de los valids y, asimismo, ser más ordenados con los archivos para simplificar el trabajo de diseño asignado y poder identificar los 'bugs' que se puedan encontrar a la hora de comprobar los módulos.

# 5. Bitácora

- Avance 1: Sofía realizó el byte unstriping y el generador de relojes. Adrián el byte unstriping. Sinaí y Meybelle terminaron el 32b a 8b y el 8b a 32b, respectivamente. Para este avance el byte striping presentó un problema con los valid de salida, dicho error se corrigió para la entrega del avance 2 y 3 el 20 de octubre. Cada persona hizo el probador y el testbench para cada módulo que se les fue asignado.
- Avance 2 y 3: Inicialmente, se repartió la carga de esta entrega en equipos de 2: Adrián y Sofía se encargaban del paralelo-serial y del phytx, mientras que Sinaí y Meybelle del serial-paralelo y el phyrx; al finalizar ambos se realizó una reunión a través de Zoom para montar el phy grande. El phyrx presentó problemas debido a la implementación inicial del serial-paralelo, por lo que el equipo 1 y el equipo 2 trabajaron juntos para solucionar este módulo. El módulo phy grande se terminó de implementar el 22 de octubre.