PROYECTO:

VEHÍCULO PESADO TELEDIRIGIDO

Grupo: PHR-G01
Espacio de trabajo:
https://upm365.sharepoint.com/sites/ProgramacionHWReconfigurable/PHR-G1

Miembros del equipo:

Álvarez Gil, Daniel(JdP)	daniel.alvarezg@alumnos.upm.es
Chen, PinZhang	pinzhang.chen@alumnos.upm.es
Mejías Cobos, Alba	alba.mejias.cobos@alumnos.upm.es
Krasimirov Ivanov, Todor	t.krasimirov@alumnos.upm.es
Antón Fernández, Daniel	d.afernandez@alumnos.upm.es

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

OBJETIVO

El objetivo principal de este proyecto es evitar, en la medida de lo posible, accidentes laborales en el ámbito de la construcción relacionados con problemas mecánicos, es decir, problemas donde un trabajador controle personalmente un vehículo pesado.

Con este proyecto se quiere evitar que un trabajador se arriesgue debido a un despiste, o por cargar más peso en el vehículo del permitido o, incluso, por un fallo en el propio sistema de la máquina.

También tiene como objetivo ser más eficaces en cuanto a las tareas que se realicen con maquinaria pesada porque si se controlan remotamente varios vehículos desde una aplicación móvil, se necesitaría menos mano de obra que si cada vehículo debe ser conducido personalmente por un único trabajador.

INTRODUCCIÓN

En ciertos ámbitos laborales, como pueden ser obras o reformas, se puede utilizar la tecnología para desempeñar tareas que puedan resultar peligrosas o monótonas para el ser humano. Por ello, se ha decidido desarrollar un proyecto que ofrezca la posibilidad de que dichas tareas estén resueltas mediante un vehículo controlado de forma remota.

El proyecto está orientado al ámbito de la construcción para el transporte de material. Inicialmente, fue pensado para que este, dada una orden mediante una aplicación móvil específica similar a un GPS de interiores para poder cargar el mapa de una habitación o planta y trazar un camino por dicho mapa para que fuese desde el origen (donde estuviese aparcado) al lugar indicado para realizar una tarea determinada y regresar a su posición inicial.

La primera idea suponía mucho mas tiempo del esperado pues hay muy pocas aplicaciones, ya existentes, que tuviesen los requisitos necesarios (poco intuitivas, incompletas…). Por ello, se decidió pasar a una segunda idea, la cual consiste en buscar una aplicación que controlase remotamente un vehículo mediante WiFi y, con lo que nos permitiese dicha "app", intentar crear un vehículo funcional que pueda ser controlado remotamente. El problema principal de esta segunda idea es que no es autónomo, el controlador (un trabajador) del vehículo debe poder ver constantemente el vehículo pues es el que realmente decide los movimientos, además de que la red WiFi tiene un alcance limitado. Si se hubiese hecho como en un inicio, con trazar el camino y definir la tarea no se necesitaría a nadie controlando.

La primera idea, además de suponer mayor tiempo de realización, supone un mayor coste ya que se necesita desarrollar una aplicación móvil específica y el vehículo capaz de conectarse a ella. La segunda, aun siendo más simple, se puede realizar en menor tiempo y se sigue alcanzando el objetivo de poder evitar tareas peligrosas para los trabajadores.

En la siguiente tabla se van a proponer las dos ideas explicadas anteriormente con los parámetros más importantes:

Número	*Descripción*	*Entidad*	*Contras*	*Pros*	*Elección*
1	*Cargar mapa de la sala y trazar el camino.*	*ETSISI*	*Caro y mucho tiempo invertido en desarrollo.*	*Eficaz e innovador.*	*NO*
2	*Control remoto simple.*	*ETSISI*	*Muy simple, poco innovador.*	*Asequible y poco tiempo de desarrollo.*	*SI*

Tabla 1: Resumen de las alternativas analizadas.

Este proyecto fue elegido porque hay numerosos accidentes en la construcción actualmente aun con todos los avances tecnológicos de estas últimas décadas, muchos de ellos por un mal manejo de la maquinaria debido a todo el peso que estas cargan.

Figura 1: Imagen de un accidente de maquinaria pesada. (El link en referencias)

Las zonas de obras están muy controladas y solo puede pasar gente autorizada por la seguridad de los ciudadanos de a pie y, aun así, muchas de las personas autorizadas a entrar (trabajadores) están expuestas a momentos muy peligrosos que pueden ser fatales. Debido a ello, se considera que este proyecto puede salvar o, al menos, menguar los daños que se causan cuando ocurre algún accidente en estos lugares.

DEFINICIÓN DEL PROBLEMA

El proyecto está enfocado a maquinaria pesada como pueden ser excavadoras, grúas, etc. El problema es que hay mucha gente trabajando en la construcción que maneja dicha maquinaria presencialmente llevando consigo materiales muy pesados y, por lo tanto, supone un riesgo en la salud de dichos trabajadores ya que cualquier error de manejo con estas máquinas puede llevar una consecuencia terrible.

Para solucionar o minimizar lo máximo posible dicho problema, los requisitos necesarios son los siguientes:

1. Aplicación de control remoto capaz de conectarse mediante WiFi al vehículo.

2. Módulo WiFi compatible con la aplicación para el correcto envío y recepción de las órdenes al vehículo.

3. Módulo FPGA para el tratado de las órdenes enviadas (información) desde la aplicación.
    3.1 Movimiento y direccionamiento del vehículo.
    3.2 Modo: pala o movimiento.
    3.3 Movimiento de la pala.

4. Sensor ultrasonido para detener el vehículo en caso de obstáculo y poder evitarse choques o atropellos.

5. Módulo de interconexión entre los elementos del sistema.

6. Montaje de un prototipo de vehículo pesado con sus correspondientes complementos para un buen funcionamiento.

    6.1 Pala para elevar y transportar carga.
    6.2 Motores para movimiento y dirección.
    6.3 Fuente de alimentación para los diferentes módulos.

DISEÑO DE LA SOLUCIÓN PROPUESTA

La solución elegida para el problema propuesto consiste en, con una aplicación móvil ya creada, controlar uno o varios vehículos a través de una red WiFi.

Primero, al ser varias personas en el proyecto, se dividieron tareas para perder el menor tiempo posible: entender el nuevo entorno donde se va a realizar el proyecto, en este caso se llama "Vivado" de *Xillinx;* leer/entender el funcionamiento de la FPGA asignada para conocer su funcionamiento; buscar una buena aplicación móvil en plataformas de *apps* (como puede ser *play store*) o internet; leer/entender cómo se implementa el módulo WiFi para conocer cuáles son los datos enviados desde la aplicación móvil y cómo mandarlos a la FPGA; y por último, conocer bien el lenguaje de programación con el que se va a implementar el código que va a soportar todo este proyecto, en este caso es VHDL.

Una vez estudiado el funcionamiento de las herramientas con las que se va a trabajar, se procede a realizar un esquema general de los módulos necesarios y su interconexión dentro de la FPGA.

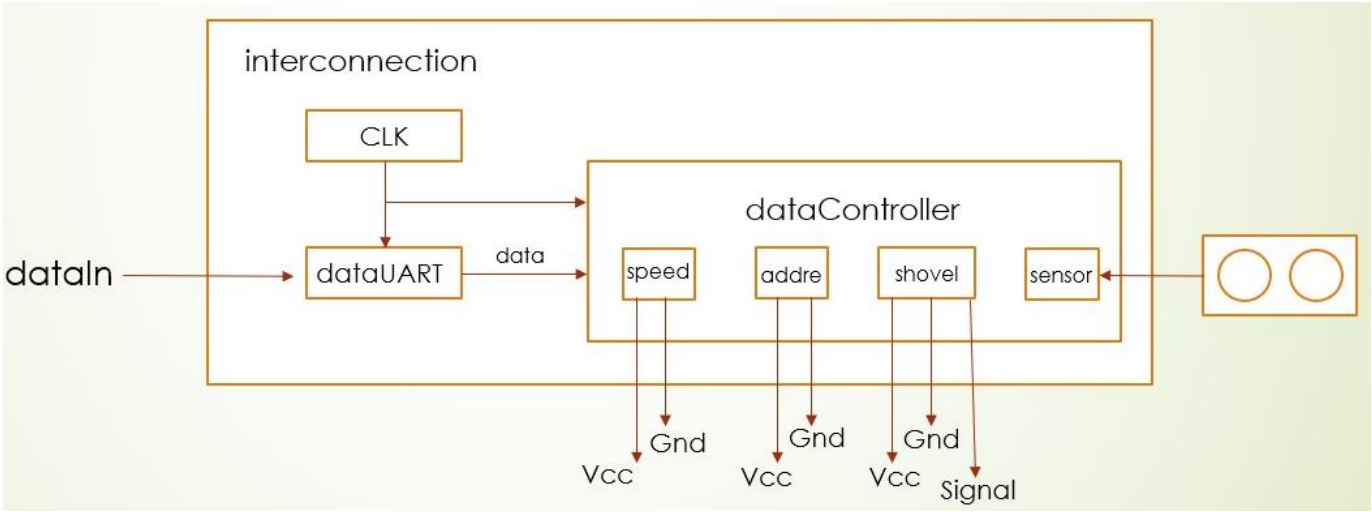


Figura 2: Esquema general.

Luego, se empieza con la implementación de los diferentes módulos y para intentar ser equitativo se repartió la tarea en 5 bloques: el módulo WiFi conectado a la aplicación móvil, el módulo *Uart* (el más complejo del proyecto), el módulo de la pala elevadora, el módulo del sensor ultrasonido y, por último, la construcción de un prototipo para una buena presentación del proyecto. Se han realizado simulaciones de las diferentes

partes y se ha comprobado el buen funcionamiento de las mismas ayudando, posteriormente, a la depuración de errores.

Más adelante, fue necesaria la implementación de nuevos módulos como pueden ser los divisores de frecuencia de algunos módulos (la *Uart* y la pala), un reloj general para sincronizar todos los componentes y registros para almacenar la información enviada de la *app* móvil.

Cuando todas las componentes están conectadas y la simulación de estas es correcta, se genera un test completo de todo el sistema para comprobar que todos los módulos cooperan entre sí. Si dicha simulación funciona como se espera, se debe cargar el código en la FPGA para ver si el funcionamiento también es correcto a nivel físico pues puede ocurrir (es lo más frecuente) que el funcionamiento en simulación sea bueno y en la FPGA no funcione nada o no como se espera.

En el caso de que haya algún error en la simulación, obviamente se debe volver a mirar la lógica del código del módulo que no funcione. Y en el caso de que en la simulación funcione y en la FPGA no, puede que el código no esté bien enfocado al mundo físico.

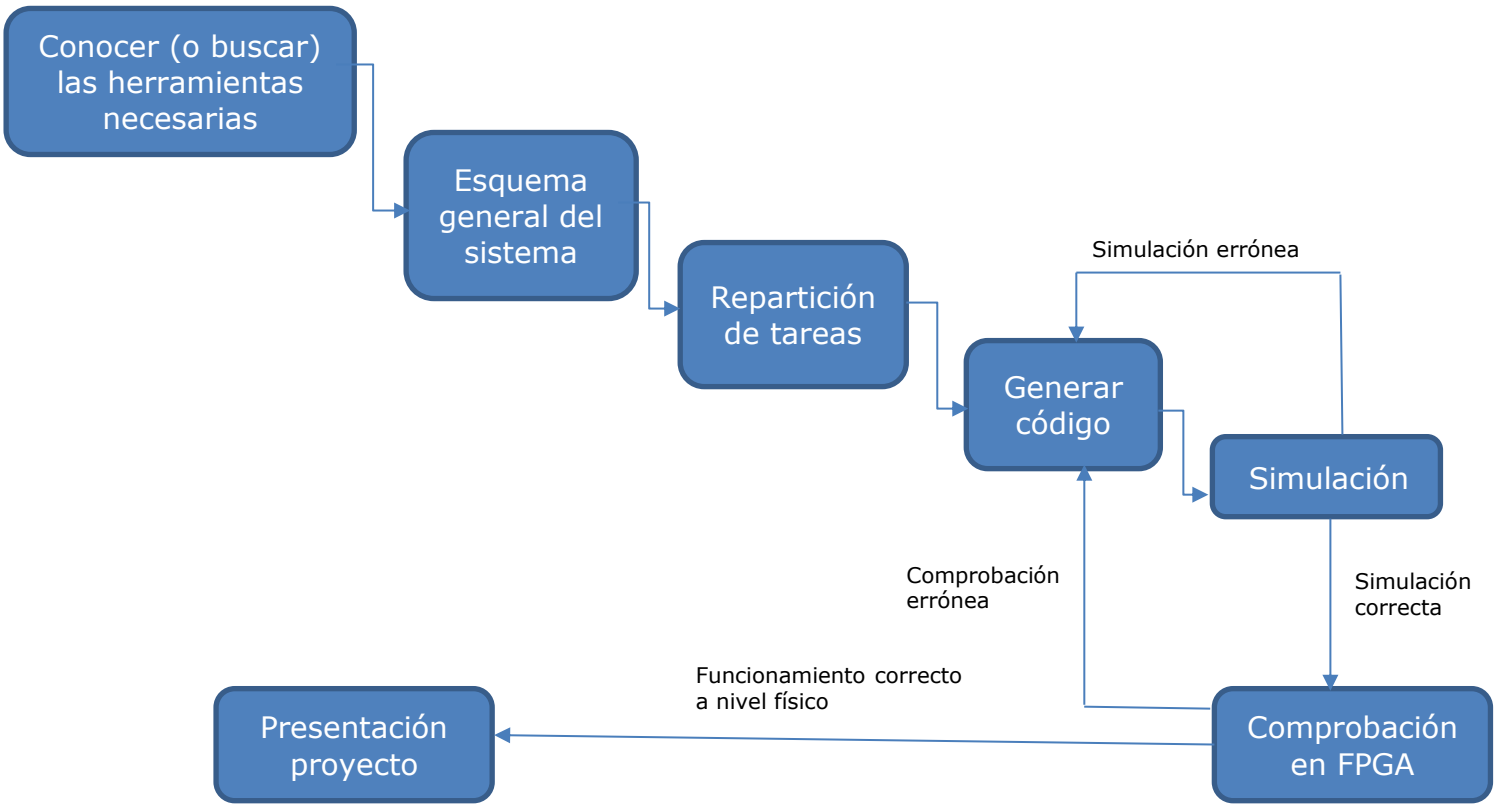


Figura 3: Esquema de ejecución del proyecto.

Por último, una vez que se ha completado el proyecto y todo funciona correctamente, se realiza una memoria explicando el objetivo, la motivación, todo lo ocurrido durante el desarrollo, las posibles mejoras, el coste del desarrollo, el material, las herramientas o entornos utilizados, etc. En general, la memoria explica todo lo que se deba saber sobre el proyecto para cualquier persona que quiera o necesite saber del tema.

HERRRAMIENTAS SOFTWARE EMPLEADAS

Vivado Design Suite es el software que ha sido empleado en ese proyecto para síntesis y análisis de diseño VHDL y también para la implementación del código en VHDL.

Paint para la generación de algunos esquemas de la memoria. Es eficaz y está disponible en cualquier ordenador.

SharePoint es la herramienta que se ha utilizado para la organización del grupo. Los documentos/ficheros realizados durante el desarrollo del proyecto se han ido almacenando en dicha plataforma, además de escribir semanalmente cada integrante del grupo lo que ha ido trabajando durante el cuatrimestre.

Arduino es el entorno de desarrollo seleccionado para poder programar el módulo WiFi. Con ello se ha conseguido una comunicación por puerto UART de la secuencia de bits que la FPGA recibe a través del módulo WiFi y que esta tiene que interpretar para realizar la acción indicada.

Word es la herramienta de texto utilizada para escribir la memoria. Se ha elegido pues es con la que el grupo se siente más familiarizado.

HARDWARE EMPLEADO

Hardware	Modelo	Precio
Coche juguete		20 €
Sensor ultrasonido	HC-SR04	3 €
FPGA	Basys3 Artix-7	133 €
Servo motor	SM-S4304R	15 €
Cables		3,10 €
Pala	Mecanos	5 €
Pilas recargables	AA Sony Cycle Energy	17 €
Módulo Wifi	NodeMCU ESP8266 v3	7 €
Chapa aluminio		2 €
		205 € = Coste Total

Tabla 2: Resumen del material.

DESARROLLO SOFTWARE REALIZADO

Los diferentes módulos implementados para el proyecto son los siguientes:

1. Aplicación móvil

La aplicación móvil será de utilidad para el movimiento del vehículo y la pala. En ella, aparecen cinco botones: ↑, →, ↓, ← y un *switch*, enviando a la FPGA 3 bits de información para que lo procese. El *switch* tiene dos modos de funcionamiento: movimiento del vehículo y movimiento de la pala.

Para el movimiento del vehículo se usan los botones de manera simbólica.

Para el movimiento de la pala, hay dos botones (→, ←) que tienen otra interpretación correspondiente a la subida y bajada de la pala respectivamente.

Para que el vehículo o la pala se paren, la propia aplicación envía una secuencia de STOP cuando se deja de presionar cualquier botón de direccionamiento.

1. Pantalla inicial **2.modo movimiento** **3.modo pala.**

Figura 4: Aplicación móvil.

Cada botón de la aplicación envía una secuencia de 8 bits diferente, las cuales son tratadas posteriormente. De los 8 bits enviados solo importan los tres últimos pues se han configurado siete posibles secuencias en nuestro mando control. Por ello, la secuencia en la aplicación va del número 129 al 135, cogiendo los tres últimos bits de estos.

Secuencia	Orden
001	Girar a la derecha
010	Hacia atrás
011	Girar a la izquierda
100	Parar
101	Modo pala
110	Modo movimiento
111	Hacia adelante

Tabla 3: Órdenes enviadas a la FPGA.

2. Módulos

2.1. Controlador de datos

Este módulo es el que trata la información que se está recibiendo desde la UART. Como se ha explicado anteriormente, la aplicación móvil nos ofrece dos modos de funcionamiento, luego cuando se decide cambiar a un modo u otro, este módulo es el que decide cuál de los motores debe funcionar y cómo.

2.2 WiFi y UART

Para tener un sistema de comunicación con la FPGA se ha utilizado una NodeMCU ESP8266 v3 que, con su módulo WiFi, mantiene una conexión entre el dispositivo y la Node. La conexión se realiza a través de un socket que con la dirección IP que se le asigna al módulo WiFi y el puerto 80 mantiene la comunicación.

Por otro lado, desde la FPGA se conecta a la Node por los puertos UART. Si desde la aplicación móvil se selecciona una acción, esta información la recibe el módulo WiFi y este se lo envía de manera asíncrona a la FPGA, la cual va concatenando bit a bit para conocer qué acción se le ha dicho que debe realizar.

2.3. Divisor de frecuencia de la UART

Como la UART recibe los datos de manera secuencial y hay un retardo entre los bits enviados, se necesita tener en cuenta dicho retardo para interpretar los datos en el

orden correcto. En este caso, es de 1Mhz para estar totalmente seguros de que se están almacenando los bits correctos.

2.4. Sensor ultrasonidos

El funcionamiento del sensor ultrasonido HC-SR04 esta descrito detalladamente en el fichero.

El módulo del sensor ultrasonidos (llamado *Top Sensor*) está compuesto por dos módulos diferentes:

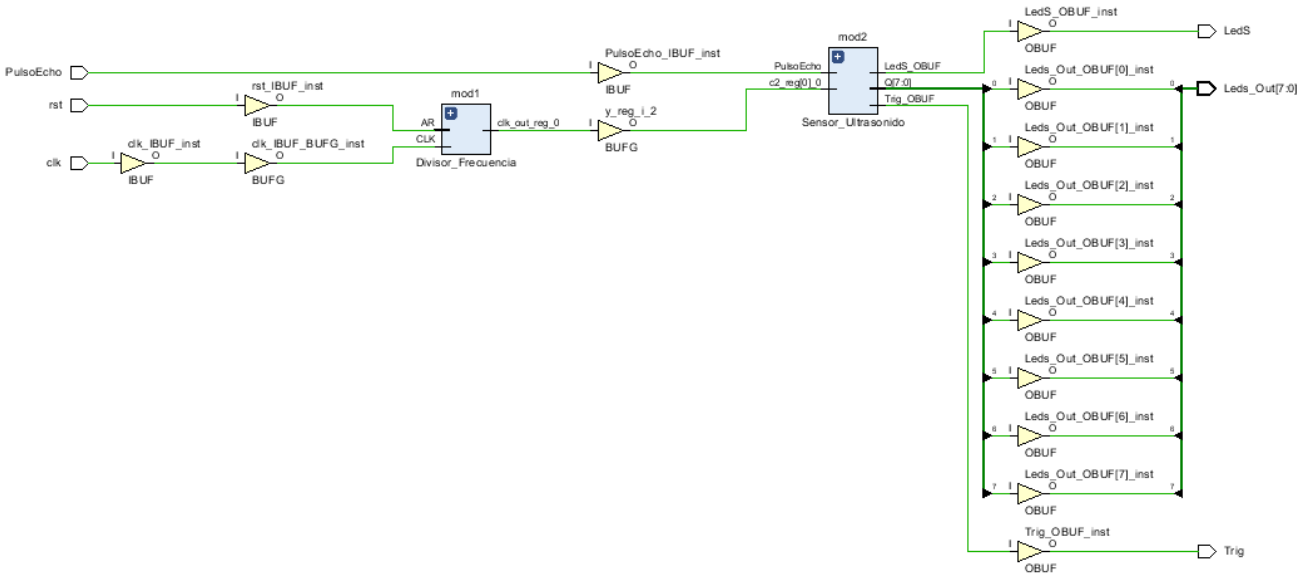


Figura 5: Esquema del módulo Sensor ultrasonido.

2.4.1. Divisor de frecuencia

Es un módulo complementario para aquellos elementos del sistema que no pueden trabajar a la velocidad del reloj general. Para este sensor, se transforma la señal del reloj a 1Mhz.

2.4.2. Detector

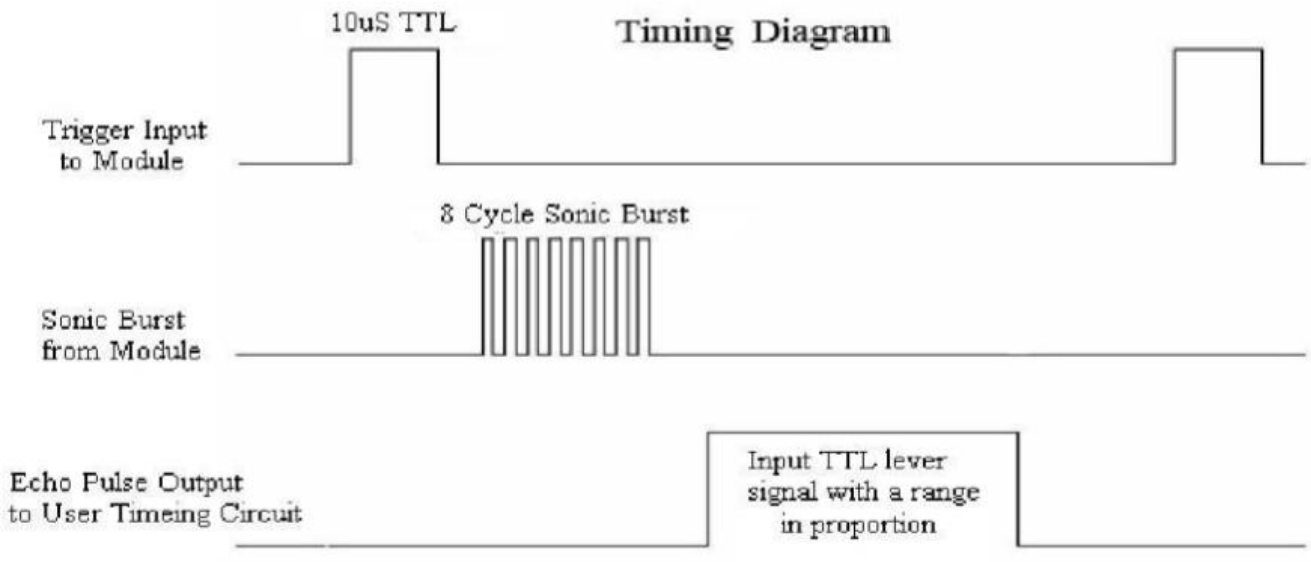


Figura 6 : Diagrama de tiempo del sensor.

Recibe la señal en un periodo de 1us y realiza la tarea correspondiente del diagrama. Calcula la distancia entre el vehículo y el obstáculo para decidir si el vehículo se para o no.

2.5. Reloj general

Para sincronizar todos los módulos de nuestro proyecto se ha utilizado el reloj interno de la propia FPGA y así, sincronizar los componentes. Es de 100 MHz, pero se ha realizado un divisor general de 1 MHz para todo el sistema.

2.6. Pala

Una vez detectado el objeto que se quiere transportar, desde la aplicación móvil pasamos a modo pala. Esto supondrá un cambio en el módulo de control de datos, derivando que dos combinaciones tengan una nueva interpretación que será para el movimiento, en el eje vertical, de la pala.

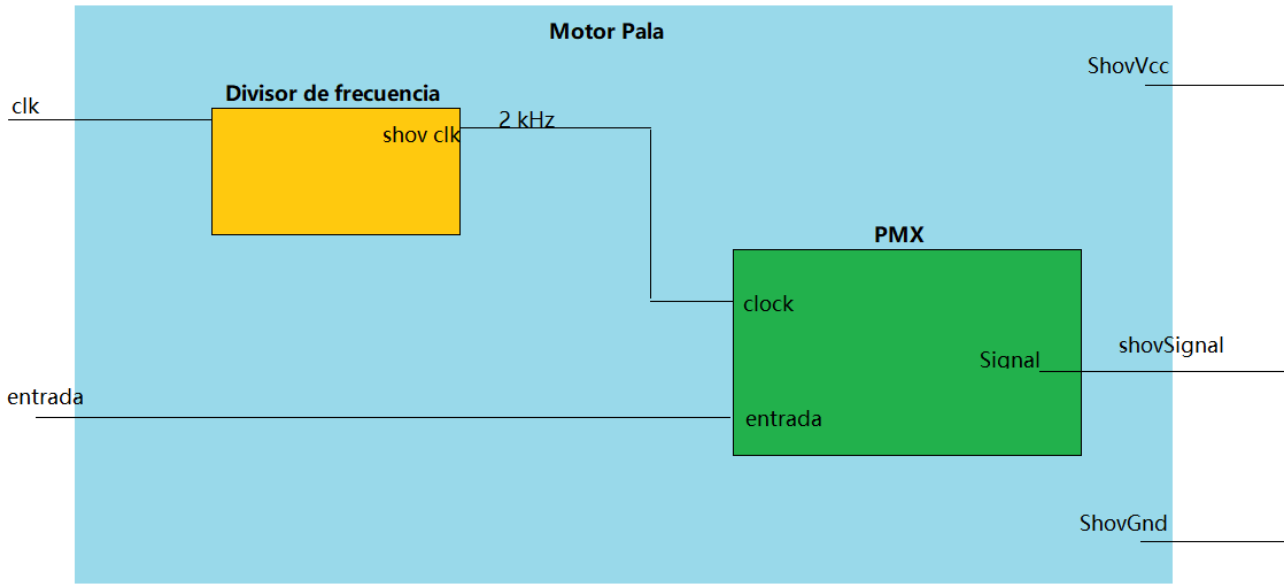
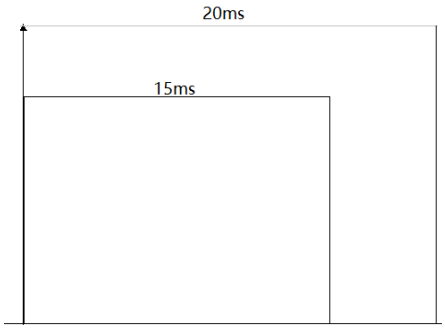


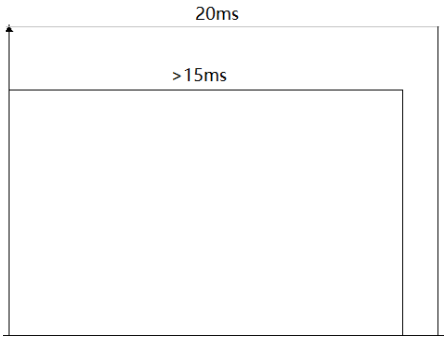
Figura 7 : Esquema de la pala (módulos).

El motor para la pala se trata de un "servo", luego funciona de una forma diferente al resto de motores. El tiempo de pulso determina el modo de funcionamiento del servo, el cual tiene un periodo de 20ms.

Modo parado



Modo horario



Modo antihorario

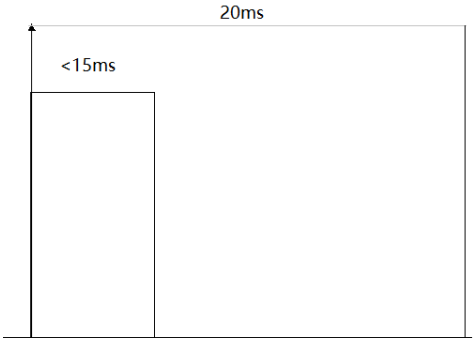


Figura 8: modos funcionamiento del motor de la pala.

Problema encontrado:
Puesto que el periodo del pulso son 20ms, durante ese tiempo se puede recibir más de una instrucción en el "Pulse Width Modulation" (PWM).

Planteamiento de la solución:
Para mayor seguridad a la hora de elevar o bajar carga, cuando en un mismo pulso de reloj (20 ms) aparece más de una instrucción, se almacena la última instrucción recibida. El mecanismo es el mismo que en el "modo movimiento", al mantener pulsado un botón de la aplicación, la pala realiza dicha operación hasta que se envía la secuencia de parada. La tabla de las órdenes de la pala es la siguiente:

Secuencia	Orden
01	Subir la pala
11	Bajar la pala
00/10	Parar

Tabla 4: Órdenes enviadas a la FPGA.

Se debe tener en cuenta que la instrucción se lee en un cierto periodo, luego se debe mantener pulsado el botón hasta que se comience a ejecutar la orden pues si se deja de pulsar en ese pulso, se enviará la secuencia de parada y la pala no se moverá.

2.7. Motores motrices y directrices.

Se ha desarrollado dos módulos por separado para el movimiento hacia adelante y hacia atrás y otro para el giro de las ruedas a izquierda y derecha.
Si el controlador de datos toma la decisión de que hay que avanzar, en función del movimiento que se ha ordenado hacer, habrá que invertir la polaridad del motor para que el vehículo se mueva hacia atrás o no invertirla para que el movimiento sea hacia adelante.
Algo similar ocurre con el motor directriz. Si se decide girar hacia algunos de los lados habrá que invertir o no la polaridad del motor para efectuar la acción.
El motor motriz se parará cuando se envié la secuencia de STOP, la cual se origina cuando se deja de presionar alguno de los botones que se esté pulsando en la aplicación. Para el directriz, el eje de las ruedas se endereza cuando recibe la secuencia de STOP.
Hay que tener en cuenta que cuando se arranca el vehículo, los valores por defecto son: el motor motriz parado y el directriz con las ruedas enderezadas.

2.8. Interconexión

Este módulo está desarrollado para generar conexión y coherencia entre los distintos módulos explicados anteriormente.

PRUEBAS Y TESTS

Prueba unitaria del **módulo sensor**:

Se crea un reloj de 100Mz para verificar el funcionamiento del divisor. Después, variar los valores de la entrada "echo" manualmente, observando los outputs "Led_Output" para comprobar el cálculo de distancia.

Resultado esperado:

- Periodo del divisor: 1 us.
- La señal de stop a '1' cuando la distancia detectada es menor o igual que 30cm.
- Led_Output: Es el intervalo de tiempo cuando echo está a nivel '1' con la siguiente fórmula: $\frac{tiempo\ en\ us}{58\ us/cm}$
- Trigger: 10 us al nivel ´1´, el resto del tiempo a nivel ´0´ hasta que alcance el tiempo máximo.

Resultado obtenido:

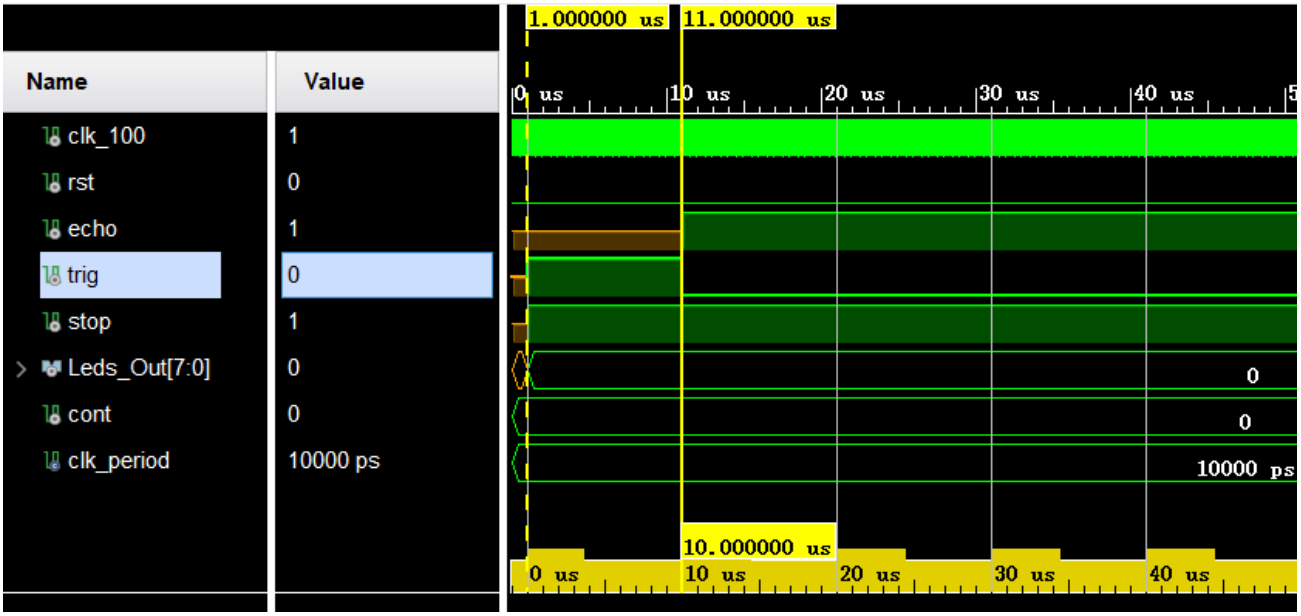


Figura 9 : Simulación correcta.

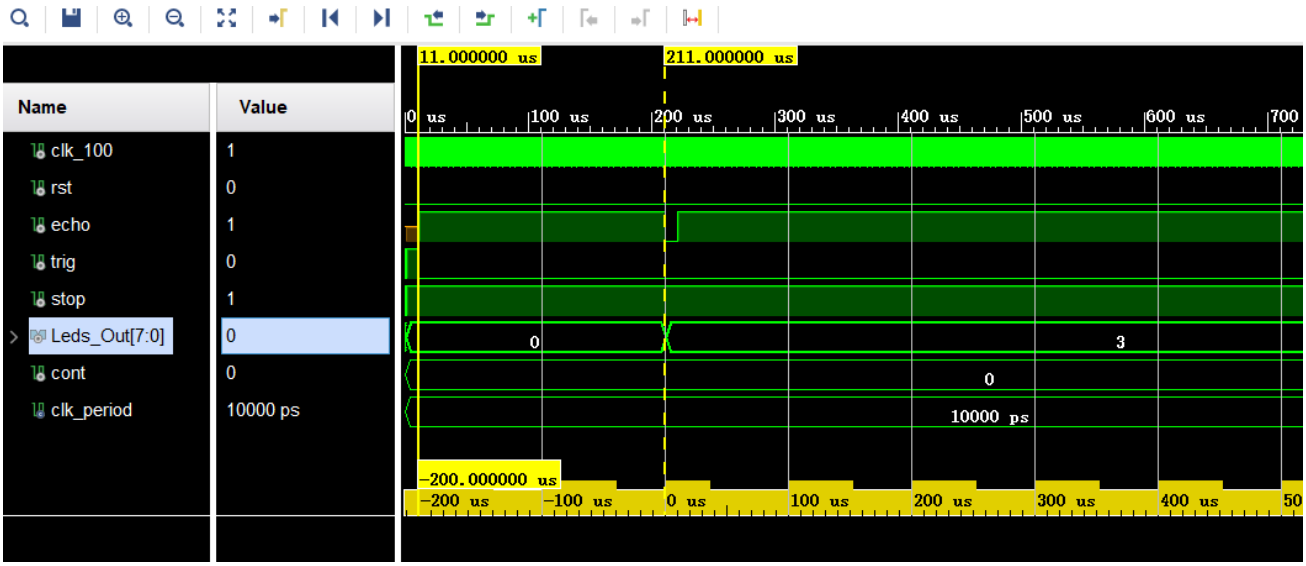


Figura 10 : Simulación explicativa.

Como "echo" ha estado a nivel '1' durante 200 us, 200/58=3.44827 cm < 30 cm, los valores de "Leds_Out" y de "stop" son correctos.

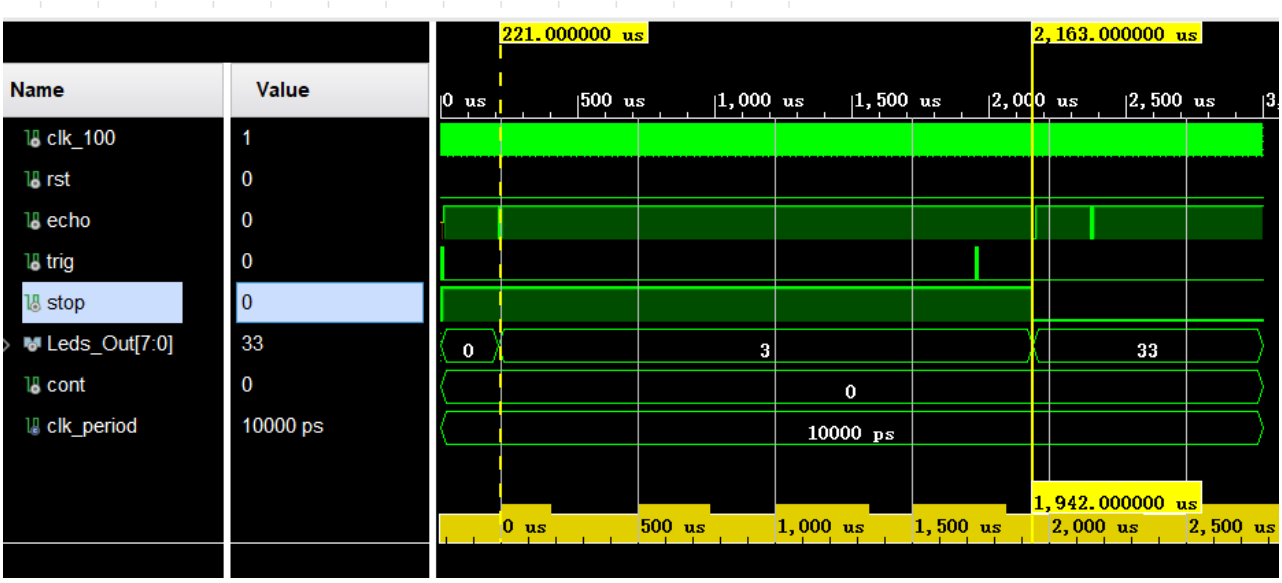


Figura 11 : Simulación explicativa.

(2163-221)/58=33.4527 cm > 30cm, se puede observar que el bit de "stop" ya es 0.

Prueba unitaria del **módulo Pala:**

Se envía una instrucción al módulo pala almacenada en el vector "entrada". Dicha instrucción se lee durante un pulso (*shovSignal*) y se decide qué acción ejecutar.

Test 1: No dar ninguna señal al vector "entrada".

Resultado esperado: No realizar tarea significativa.

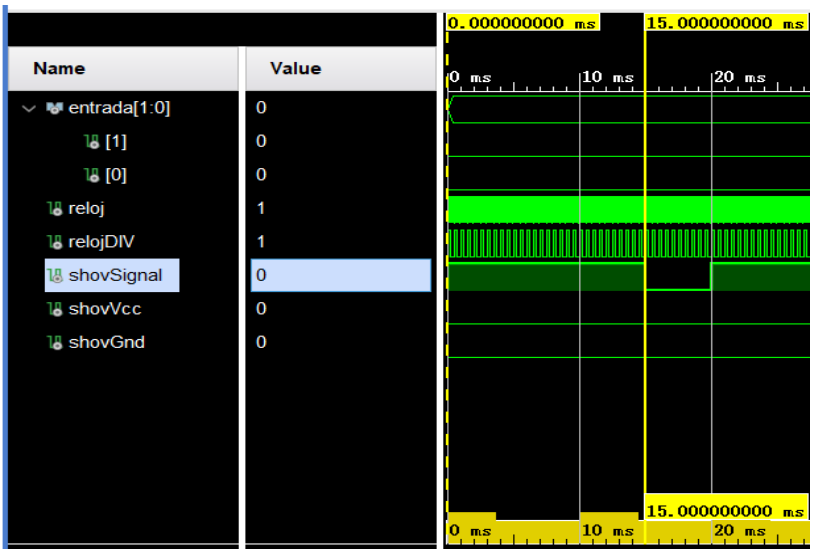


Figura 12 : Simulación explicativa de Test 1.

Resultado real: El motor esta parado siempre y *shovSignal* ejecuta la instrucción que se recibe, en este caso es "00" (durante los 15ms del pulso) pues no se está enviando nada y por defecto la pala está parada.

Test 2: Dar instrucciones al inicio de cada periodo de PWM.

Resultado esperado: Ejecutar la instrucción recibida.

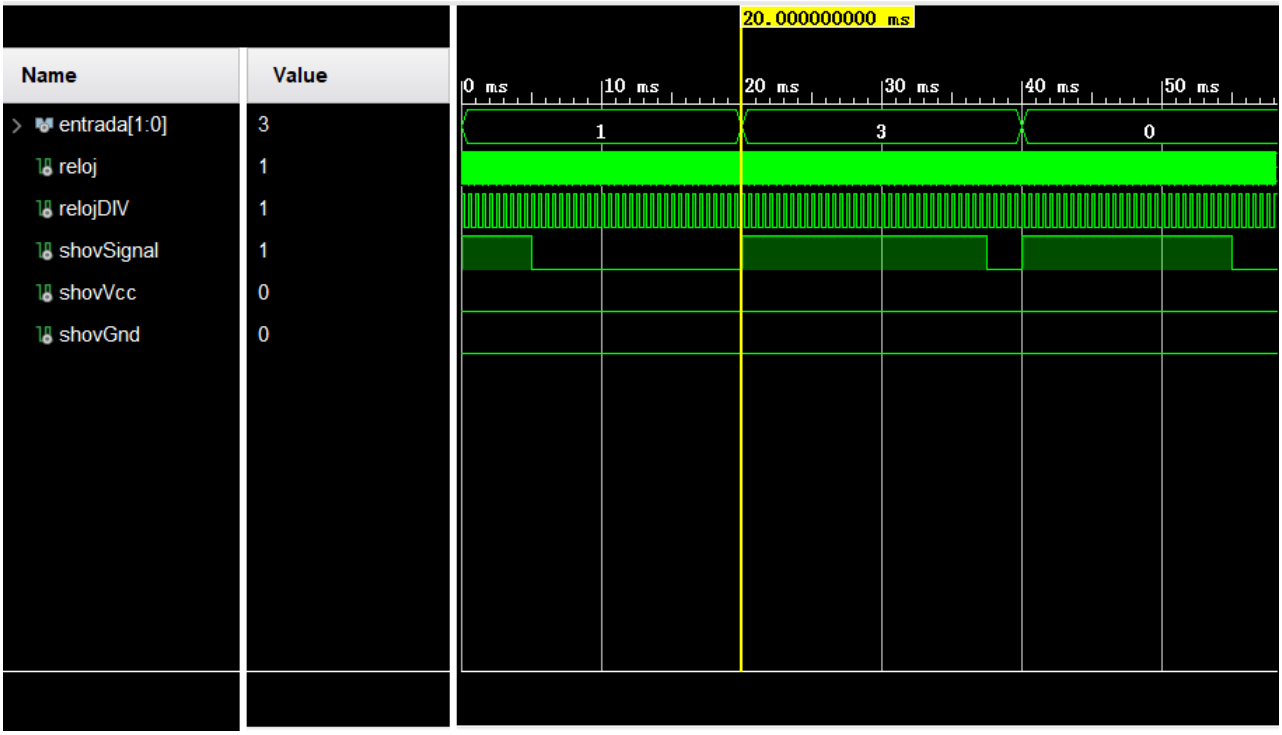


Figura 13 : Simulación explicativa de Test 2.

Conociendo la tabla de instrucciones explicada en el módulo de la pala, se puede observar que se está leyendo la instrucción "01" y se ejecuta durante 5ms (subida de la pala). Luego, la instrucción "11" se recibe justo al inicio del siguiente pulso y se ejecuta durante mas de 15ms (bajada de la pala). Por último, en el siguiente periodo, se ejecuta la instrucción de parada se ejcuta durante 15 ms.

Test 3: Test de sobrecarga.

Consiste en dar varias instrucciones en el mismo periodo de PMX.

Resultado esperado: En el dicho periodo, realiza la tarea de la primera instrucción recibida, y en el siguiente periodo, realiza la tarea de la última instrucción recibida.

Resultado real:

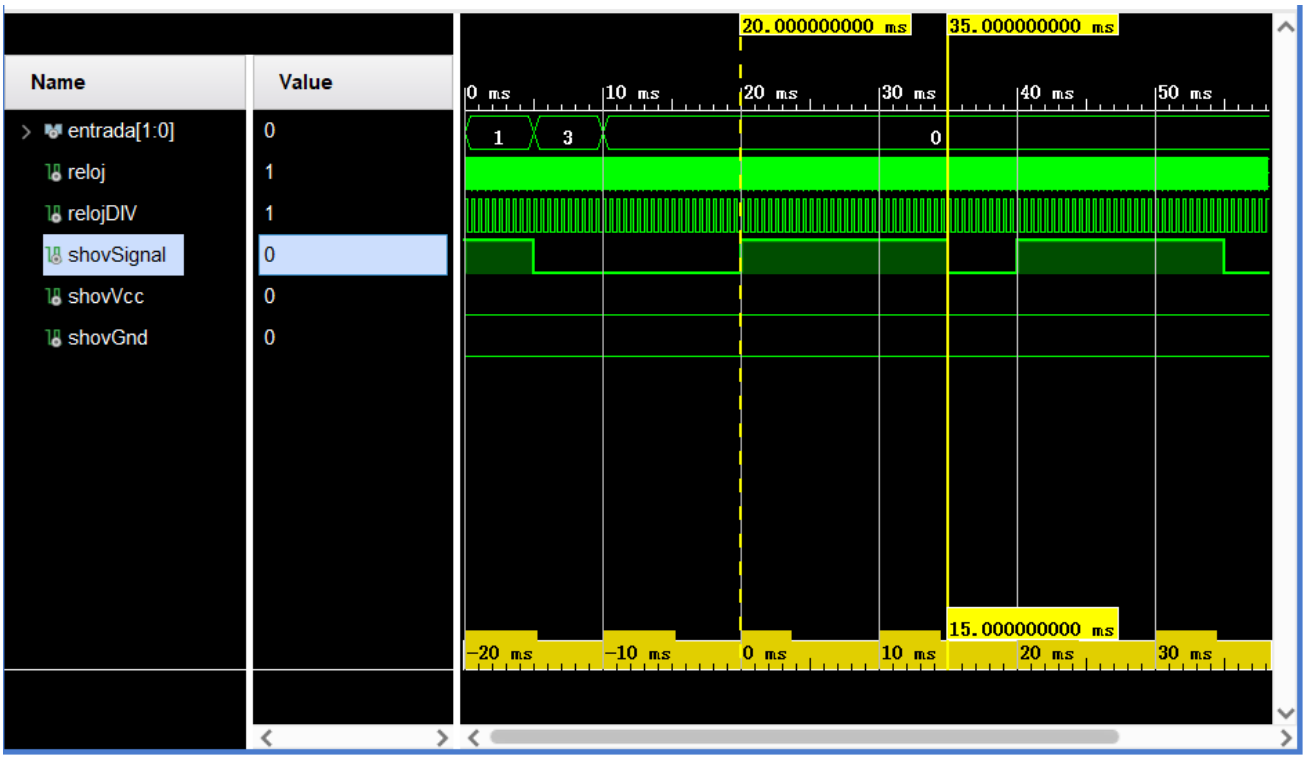


Figura 14 : Simulación explicativa de Test 3.

La primera instrucción recibida es "01" (5 ms de pulso) y la última instrucción del mismo periodo es "00" (15ms de pulso).

Se puede observar que, en el primer caso, no se cambia o sobrescribe la tarea en el mismo periodo una vez ejecutada "01", sino que, ya en el siguiente pulso, se ejecuta la instrucción que se reciba en dicho momento, en este caso "00" (parada).

PLANIFICACIÓN Y COSTES

En el anteproyecto se realizó la siguiente tabla con los hitos que se iban a realizar y el tiempo que iba a conllevar:

NÚMERO DE HITO	DESCRIPCIÓN	FECHA
1	Resolver comunicaciones: Enlazar la comunicación entre una aplicación que determine la ruta y la FPGA.	29-4-2019
2	Montaje electrónico del coche.	29-4-2019
3	Programación en VHDL del cálculo de movimientos.	6-5-2019
4	Integración de los hitos anteriores.	14-5-2019
5	Detección de obstáculo y envío de notificación.	16-5-2019
6	Programación para deshacer la ruta y regresión.	20-5-2019
7	Realización de tarea.	
8	Aparcamiento tras la vuelta.	

Tabla 5: Hitos anteproyecto.

Los hitos en rojo son aquellos que se iban a realizar si se daba bien el proyecto y sobraba tiempo.

A continuación, se va a mostrar la tabla de hitos desarrollados en el proyecto final:

NÚMERO DE HITO	Integrante del grupo	DESCRIPCIÓN	FECHA	Finalizado	Motivo
1	D. Álvarez	Enlazar la comunicación entre una aplicación móvil y FPGA.	20-05-19	SI	No se sabía lo que realmente enviaba la aplicación.
2	T. Krasimirov A. Mejías	Programación en VHDL del tratamiento de los datos.	-		Problemas con almacenamiento de la información.
3	P. Chen	Detección de obstáculo y envío de notificación.	15-05-19	SI	Simulación correcta.
4	P. Chen A. Mejías D. Antón	Programación de la pala.	-	SI	Simulación correcta.
5	T. Krasimirov A. Mejías D. Álvarez P. Chen D. Antón	Integración de los hitos anteriores.	-	NO	Debe funcionar todo el sistema.
6	D. Antón	Montaje electrónico del coche.	20-05-19	SI	Se fue modificando la estructura para mayor comodidad.

Tabla 6: Hitos finales.

Como se puede apreciar en ambas tablas, se han tenido que dejar algunos hitos pendientes pues no ha dado tiempo a todo lo que se pretendía hacer, como por ejemplo los hitos en rojo. Sin embargo, no ha habido grandes desviaciones ya que sigue siendo el mismo proyecto pero algo más simple.

Las fechas de finalización son orientativas porque la gran mayoría se han ido modificando hasta la presentación del proyecto.

El coste total del proyecto, teniendo en cuenta el trabajo de los 5 integrantes y el HW necesitado. Además, se ha trabajado unos 2 meses sobre el proyecto, por lo que se va a calcular un salario medio al mes por cada integrante para calcular los costes "personales".

Costes Totales = (prototipo) 205 + 1.500*5(integrantes)*2(meses) = 15.205 €

El diagrama de *Gantt* del proyecto es el siguiente:

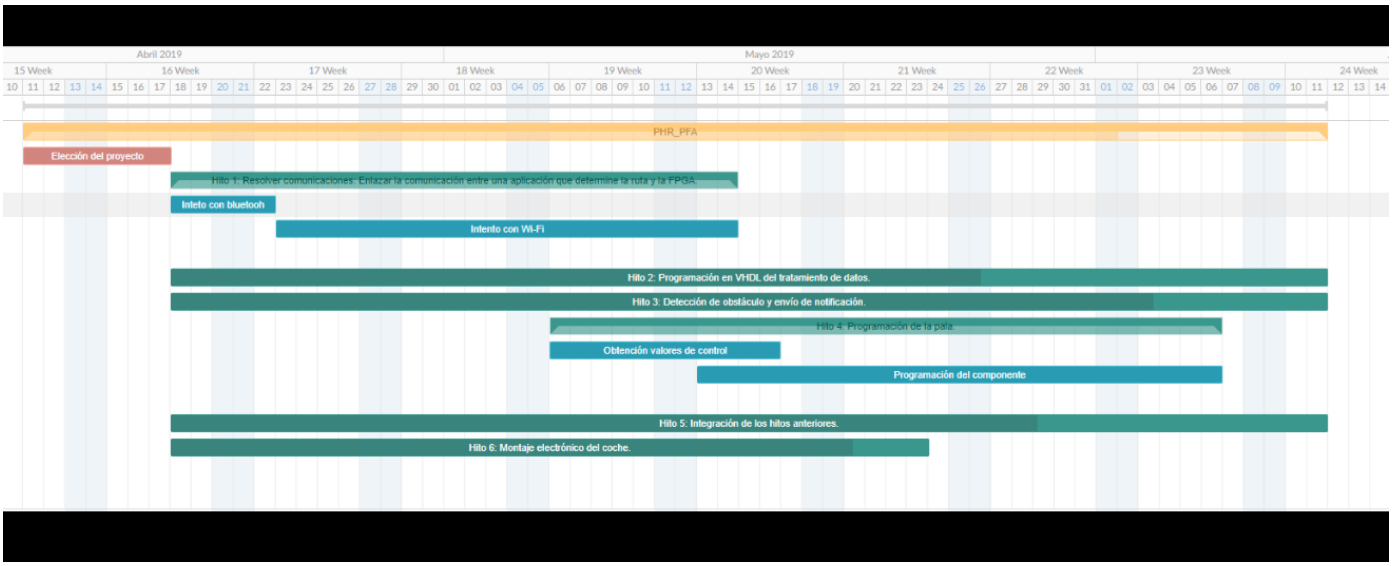


Figura 15 : Diagrama de *Gantt*.

ASPECTOS SOCIALES, AMBIENTALES, ÉTICOS Y LEGALES

Al ser máquinas muy pesadas y de gran volumen, consumen mucho combustible y, además, la gran mayoría de esta maquinaria funciona con combustibles fósiles por lo que tiene un gran impacto medioambiental.
Se debe mencionar también el impacto audiovisual que provocan dichas máquinas durante el tiempo que se encuentran en una obra, sobre todo en zonas muy pobladas como pueden ser obras en el centro de las ciudades. Además, tal como indica el periódico "El Diario", España lleva siendo durante tres años el país con más infracciones ambientales de la Unión Europea. Se debería empezar a concienciar sobre este problema cuanto antes y tomar medidas. Sobre el tema que nos concierne en este proyecto, podría comenzar a haber algunos de estos vehículos alimentados con energías renovables, aunque no sean tan potentes ni aguanten tanto tiempo funcionando.

Dentro del marco social puede suponer un problema de empleabilidad pues, según se va dando mayor autonomía a las propias máquinas, más prescindible es el trabajador de a pie, en el ámbito de la construcción lo que se llaman peones. Aunque es cierto que aumentaría el trabajo en el sector del mantenimiento de maquinaria.

El aspecto ético está más enfocado a los propios trabajadores involucrados en el ámbito de la construcción ya que son ellos los que tendrían la posibilidad de utilizar esta aplicación. Sin embargo, se presupone que no todos los trabajadores podrían utilizarla, sino solo aquellos responsables que conozcan la aplicación móvil y entiendan lo que conlleva realizar una buena orden a la máquina para evitar accidentes que perjudiquen a algún trabajador o a la obra que se estuviese desarrollando.

Al ser una aplicación que controla remotamente un vehículo, mientras que esté bien controlado dentro del perímetro de la obra y no ocasione un obstáculo para la vía pública, no se considera que tuviese ninguna consecuencia legal. Por lo que se ha investigado, el control remoto no se ha llevado a un vehículo pesado, es decir, solo se ha podido encontrar noticias y foros sobre vehículos pesados a escala luego puede que no haya una ley aun que especifique la línea roja en este tema.

Aun así, si se quiere cargar el mapa de una zona concreta, debido a la nueva ley de protección de datos, es posible que haya que pedir permiso (en el contrato de la obra o reforma, por ejemplo) al dueño del terreno y explicar para qué se va a utilizar el escaneo de la zona en el que se va a trabajar.

CONCLUSIONES

Durante todo el tiempo que se ha desarrollado el proyecto no hemos tenido ningún problema en la parte de simular los diferentes elementos del sistema. Sin embargo, a la hora de cargar el código en la FPGA y probar a nivel físico no ha habido éxito, y no ha sido por intentarlo.
Desde el momento en que se planteó la idea de que quizás no se estaba enfocando bien el proyecto a un nivel físico, se comenzaron a hacer pruebas unitarias de cómo almacenar la información que se recibe desde la aplicación ya que es el problema que nos ha perseguido siempre, además de leer por internet con el objetivo de intentar llegar a alguna solución.

Se han desarrollado cantidad de módulos VHDL para conseguir el funcionamiento correcto del puerto serie UART. Con el fin de no copiarlo de internet, se fueron implementando ideas buenas desde el punto de vista funcional. Ninguna de ellas ha sido satisfactoria a nivel físico, aunque siempre correctas cuando de una simulación se trataba.
Unos días antes de la entrega del proyecto, se comprendió cómo funciona realmente un lenguaje de descripción hardware, lo cual desencadenó una modificación exhaustiva de todo el software con el objetivo de conseguir el funcionamiento a nivel físico. Debido a la falta de tiempo, no ha sido posible esta alternativa tan arriesgada como necesaria. Se ha llegado a avanzar mucho desde ese momento, pero no se ha logrado terminar el proyecto con un comportamiento 100% correcto. Dos días antes de la entrega del proyecto, se decidió tomar el código del puerto UART de internet, que es el problema principal surgido, pero había demasiadas dependencias que se han hecho inabordables con el poco tiempo restante. Un ejemplo es conseguir una sincronización perfecta del reloj lector de datos entrantes, la definición de una señal de reloj mediante VHDL (la descripción de relojes no se implementa), disponibilidad de todo el equipo necesario para las pruebas… Elementos cuya falta impide funcionamiento alguno.

A pesar de todos los problemas surgidos, se puede concluir que es un proyecto interesante que puede ofrecer muy buenas utilidades y, además, la flexibilidad de funcionalidades es un punto a favor, pues te permite reprogramar aspectos y modelarlos a tu gusto.

LÍNEAS FUTURAS

Es un proyecto relativamente simple y, por ello, pueden realizarse numerosas mejoras para que sea un vehículo más autónomo y eficiente.

Se ha comentado la idea inicial del proyecto, la cual es bastante más compleja y no se ha podido realizar por diversas razones. Sin embargo, en lugar de obtener una aplicación externa e intentar aprovechar todo lo que ofrezca, se podría desarrollar una aplicación propia para estos vehículos pesados con las diferentes tareas que puede realizar cada uno, es decir, dependiendo del vehículo que se controle en el momento que pueda hacer unas tareas u otras ya que cada máquina tiene unas funciones y una estructura concreta.

Como se ha dicho anteriormente, las zonas de obras suelen estar perimetradas y vigiladas, ya que para un realizar un buen trabajo se debe tener muy controlado el material, la maquinaria, el personal autorizado, etc. Este factor es una gran ventaja porque a la aplicación móvil a desarrollar se le puede añadir, además de un control remoto simple como el que se ha realizado en este proyecto, una nueva funcionalidad para dar mayor autonomía: cargar el mapa de la zona de trabajo (en esta caso una obra determinada). Con un mapa de la zona de trabajo es posible trazar un camino por él de modo que el vehículo sepa en qué posición del mapa se encuentra, hacia dónde debe dirigirse y qué debe hacer una vez llegado al destino. Esta autonomía es bastante útil ya que actualmente se debe tener a un trabajador disponible para controlar la máquina. Si por el contrario, un trabajador con esta aplicación móvil puede ordenar una tarea y el propio vehículo es capaz de ir al destino y realizarla, puede conseguirse una mayor agilidad en el proceso de construcción porque puede aprovecharse ese tiempo de manejo de maquinaria en otros temas.

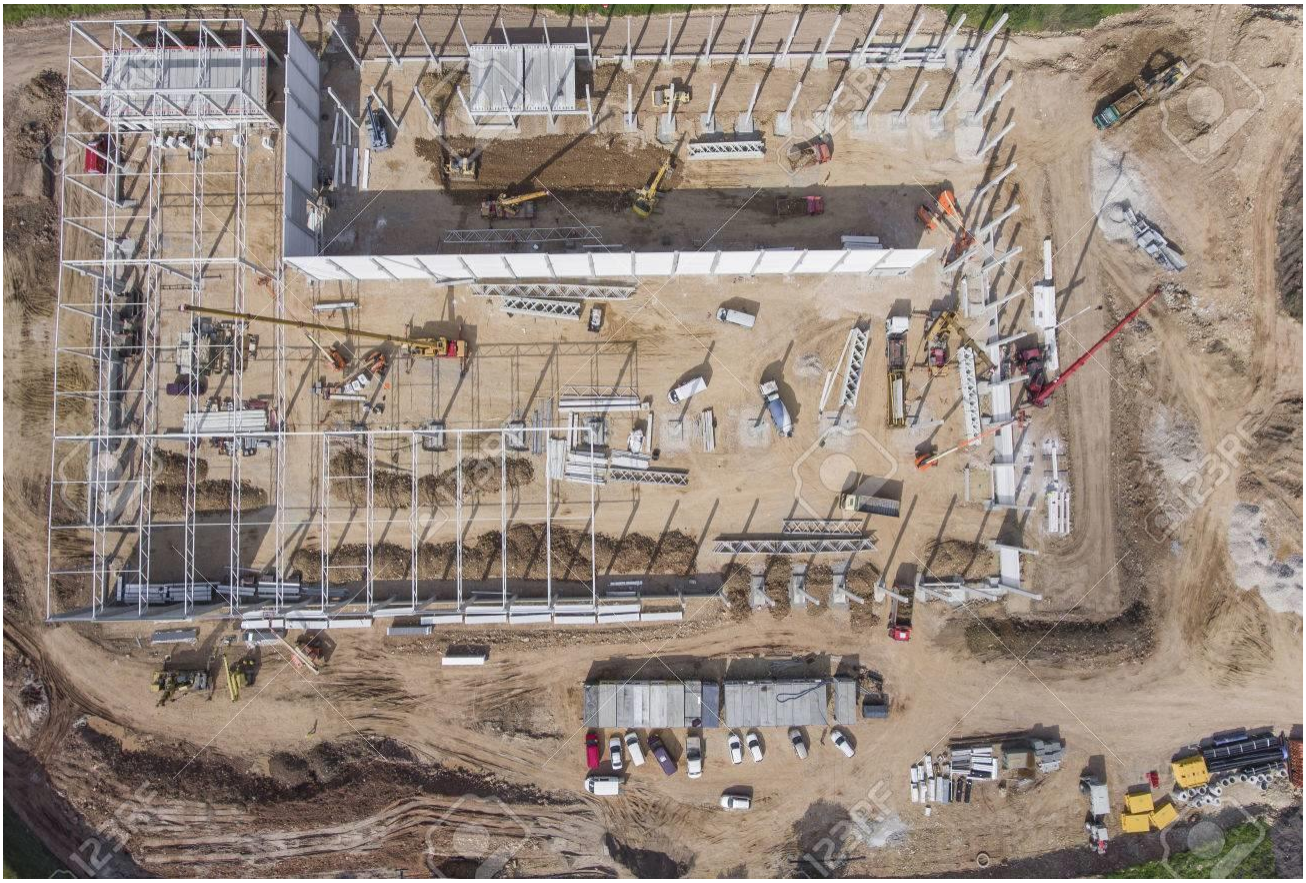


Figura 16 : **mapa de una zona de construcción desde arriba. (El link en referencias).**

Otra funcionalidad a la *app* puede ser que, una vez la máquina realice la tarea ordenada, vuelva a su posición inicial. Esto se debe a que una zona de trabajo debe estar despejada y tener a diferentes máquinas paradas en medio sin una tarea que hacer no es eficiente. Por ello, tener una zona específica en la zona de trabajo (posiblemente cargada en el mapa) donde las máquinas puedan volver cuando han terminado su labor es un trabajo mucho más limpio.

REFERENCIAS

(s.f.). Recuperado el 27 de 04 de 2018, de Imagen predeterminado de Microsoft.: http://microsoft.com

(2011). *IEEE Standard VHDL Language Reference Manual - Redline.* publisher not identified.

Xilinx. (26 de 04 de 18). Obtenido de http://www.xilinx.com

Plataforma web **Moodle UPM** donde se encuentra el material necesario para el desarrollo de la asignatura: lenguaje VHDL y teoría. Obtenido de https://moodle.upm.es

La aplicación utilizada para el control del vehículo se llama "**Wifi RC Car ESP 8266**" y se encuentra en *play store* en el siguiente *link*: https://play.google.com/store/apps/details?id=com.lacour.vincent.wificaresp8266&hl=es_419

LINKS IMÁGENES

Imagen 1. Maquinaria pesada:

https://www.rinconabstracto.com/2012/04/maquinaria-pesada-en-problemas-

fotos.html

Imagen 16. Mapa de una construcción desde arriba: https://es.123rf.com/photo_62231853_obra-de-construcci%C3%B3n-dispar%C3%B3-desde-arriba-.html

DOCUMENTACIÓN EXTERNA

PDF de la facultad de informática UCM: intro_VHDL.pdf

Noticia medio ambiente: https://www.eldiario.es/sociedad/Espana-infracciones-ambientales-Union-Europea_0_860214702.html

Noticia control remoto de vehículos: https://www.xataka.com/vehiculos/el-jefe-de-la-policia-de-londres-querria-tener-control-remoto-sobre-todos-los-coches-de-esa-ciudad

Implementar divisor de frecuencia: https://www.fpga4student.com/2017/08/vhdl-code-for-clock-divider-on-fpga.html

Implementar sensor ultrasonido: https://stackoverflow.com/questions/27580741/vhdl-ultrasonic-sensorhc-sr04

Datasheet HC-SR04: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

Datasheet Basys-3: https://reference.digilentinc.com/basys3/refmanual

Puerto UART:
https://www.digikey.com/eewiki/pages/viewpage.action?pageId=59507062#UART(VHDL)-CodeDownload

ANEXOS

Anexo I. Material entregado en sharepoint

La entrega se realiza en la carpeta ***PFA*** en la documentación de Sharepoint.

- **Prueba_y_codigo_pala.rar**
  Proyecto Vivado con módulos VHDL en los que se describe el comportamiento de la pala montacargas, con su respectivo fichero de pruebas.
- **Sensor_Mejorado_con_testbench.rar**
  Proyecto Vivado con módulos VHDL donde se ha descrito la utilidad del sensor ultrasonido. Dispone también de un fichero de pruebas.
- **Proyecto.rar**
  Proyecto Vivado que contiene la estructura principal del proyecto. No se encuentra funcional pero la descripción es la más acertada posible según lo dicho en la sección *Conclusiones*. Cabe comentar que existen varios proyectos con implementaciones diferentes para el mismo problema pero que han quedado descartados por esta versión.
- **puertoUart.vhd**
  Es la descripción propia que se ha realizado del tratamiento de datos en el puerto UART, no incluida en el proyecto final.
- **NodeMCU-WiFi_UART_V2.ino**
  Es el fichero con el que se ha programado el módulo WiFi *NodeMCU* para que enviara por el puerto UART la combinación deseada, recibiendo los datos desde la aplicación móvil.