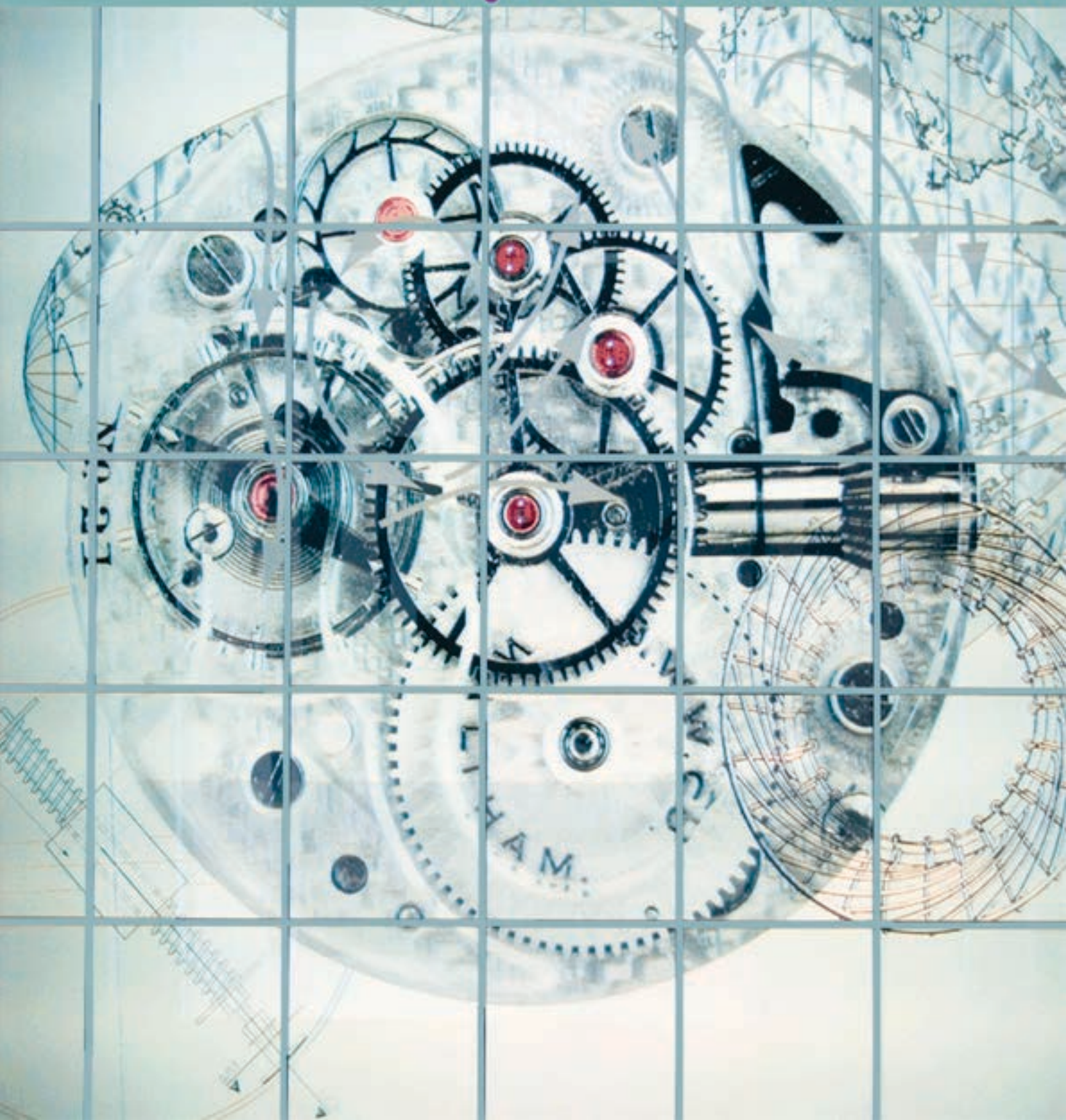
enginy@eps

Revista de l'Escola Politècnica Superior
Universitat de les Illes Balears
Número 1, curs 2007-2008

Projecte Motiva't, Motiva'l



enginy@eps

Resultat del projecte d'innovació docent: "Recerca de la Motivació de l'estudiant a les carreres tècniques: La Valoració Social i l'Expectativa d'Èxit"

Edició amb el suport de l'institut de Ciències de l'Educació

Imatge portada: Driving Force, Stuart Keeler, www.keeler-machnic.com

L'Escola Politècnica Superior de la UIB imparteix els estudis d'Arquitectura Tècnica, les engiyeries tècniques en Informàtica de Gestió, Informàtica de Sistemes, Telecomunicacions (especialitat Telemàtica), Industrial (especialitat Electrònica Industrial), així com la llicenciatura de Matemàtiques i l'Enginyeria Informàtica.

Al llarg de curs 2007-08, un grup de professors ha treballat amb il·lusió per fer un recull de pràctiques i projectes representatius dels nostres estudis. S'han triat, doncs, treballs de qualitat i que alhora servissin per mostrar a tothom els coneixements i disciplines impartits a l'Escola.

Els protagonistes són, no hi ha dubte, els alumnes que han participat en aquest projecte educatiu per mitjà del quan han tingut l'al·licient addicional de veure com el fruit del seu esforç prenia la forma d'una publicació i arribava més enllà de les parets de l'aula, el laboratori o el recinte universitari.

Com a director de l'Escola Politècnica Superior, vull agrair sincerament el treball de tota la gent que ha fet possible que aquesta publicació fos una realitat. Ara, el meu desig és que l'experiència es pugui repetir en els propers cursos i que la difusió de la tasca d'alumnes i professors faciliti una mica més l'acostament de l'Escola a la societat que ens envolta.

Gabriel Oliver Codina
Director de l'EPS de la UIB

Aquesta revista és fruit de la recerca feta dins el marc del projecte d'innovació docent anomenat "Recerca de la Motivació de l'estudiant a les carreres tècniques: La Valoració Social i l'Expectativa d'Èxit" i finançat per l'Institut de Ciències de l'Educació de la UIB. Aquesta recerca tenia per objectiu, explorar un element considerat important per a un aprenentatge eficaç: la motivació de l'estudiant.

Segons T. Jenkins la motivació és un concepte molt complexe i difícil de quantificar i promoure. A més, allò que motiva a un estudiant pot desmotivar a un altre. Així mateix, la motivació d'un estudiant és funció de dos factors: la valoració i l'expectativa d'èxit que es té de la feina que s'està fent. Així, un estudiant ha de valorar positivament l'aprenentatge que rep, però també ha de tenir expectatives d'un final positiu de l'aprenentatge. Aquesta revista que ara teniu a les mans ha servit per a que l'estudiant de les assignatures participants en el projecte valori positivament el fet d'aprendre a escriure de forma correcta sobre la feina feta, però també introduir expectatives en veure la seva feina publicada i per tant reconeguda fora de l'aula.

La realització d'aquest projecte durant el curs 2007-2008, ens ha duit a pensar que la revista seria una bona eina educativa i de motivació externa tant pels alumnes que cursen les matèries com pels alumnes que no les cursen. Permet l'utilització de l'escriptura com a eina útil per aprendre els coneixements d'una matèria per part dels autors, però també, la seva lectura serà útil per a desenvolupar competències generals de comprensió i d'utilització d'elements en un format tècnic (taules, gràfiques, diagrames).

Voldria agraïr la participació dels estudiants de les diferents assignatures participants en el projecte, així com especialment als professors que han adaptat les planificacions docents, realitzant trobades de coordinació i participat en totes les tasques de la publicació d'aquesta revista.

Acull la revista veient les possibilitats que té i transforma les mancances en fites per a la millora!

Bartomeu Alorda Ladaria

Índex d'articles

Diseño, simulación y experimentación de un circuito acoplador direccional de tres secciones

Ivan de Paúl Bernal, Alejandro García Coll

Circuitos de Alta Frecuencia
ivan.depaul@uib.es, xela1975@gmail.com

Resumen— **En este documento se muestra la propuesta de diseño y simulación de un circuito acoplador con el software ANSOFT DESIGNER. El diseño propuesto para esta experimentación es el de un acoplador direccional de 20dB de tres secciones con una respuesta binomial, una impedancia del sistema de 50Ω y una frecuencia central de 3GHz. Partiendo de los resultados del simulador, se realizará el diseño sobre una PCB. Además, se presentará la experimentación en el laboratorio analizando los resultados obtenidos en el simulador. Para ello se obtendrán los parámetros S de reflexión de los cuatro puertos y se medirá la potencia acoplada.**

I. INTRODUCCIÓN

Un acoplador direccional es una red pasiva de cuatro puertos donde uno de los puertos queda aislado respecto al puerto de entrada (*isolated port*). De los otros dos puertos, uno es el que recibe la mayor parte de la señal incidente (*through port*) y el otro el que recibe una parte fija de la señal (*coupled port*). Dicho de otra manera, es una red de cuatro accesos que tiene dos pares de accesos desacoplados. Esto implica que la matriz de parámetros S presenta cuatro elementos que no pertenecen a la diagonal principal nulos. Además, la red es recíproca y se considerada idealmente sin pérdidas.

Básicamente, consta de dos líneas de transmisión y un mecanismo de acoplo entre ellas. Un esquema del dispositivo es el que se muestra en la Figura 1. Suponiendo los cuatro puertos cargados con sus impedancias características, y a la frecuencia de diseño del acoplador:

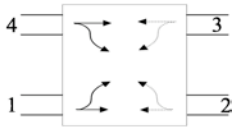


Figura 1. Esquema de un acoplador direccional

- Una porción de la onda que viaja de 1 (puerto de entrada) a 2 (puerto directo respecto al puerto 1) se acopla a 3 (puerto acoplado respecto al puerto 1) pero no a 4 (puerto aislado respecto al puerto 1).

- Una porción de onda que viaja de 2 (puerto de entrada) a 1 (puerto directo respecto al puerto 2) se acopla a 4 (puerto acoplado respecto al puerto 2) pero no a 3 (puerto aislado).

- De forma análoga para los puertos 3 y 4.

Además, los cuatro puertos están perfectamente adaptados a la frecuencia de diseño. Es decir, si tenemos tres puertos terminados con sus impedancias características, el coeficiente de reflexión a la entrada del cuarto puerto es nulo. Además, se trata de un circuito recíproco, por lo que su matriz de parámetros S o de distribución es simétrica (i.e., $S^T = S$).

Los parámetros básicos a determinar a la hora de caracterizar el funcionamiento de un acoplador direccional de potencia son el acoplo, la directividad y el aislamiento. Suponiendo los puertos 2, 3 y 4 cargados por sus impedancias características y un generador de impedancia interna igual a la impedancia característica del puerto 1 conectado en dicho puerto:

- El acoplo, C(dB), se define como

$$C(dB) = 10\log\left(\frac{P_1}{P_3}\right) = 10\log\left(\frac{1}{|S_{31}|^2}\right)$$

siendo P1 la potencia incidente en el puerto 1 (potencia de la onda progresiva que se propaga por ese acceso) y P3 la potencia que sale por el puerto 3.

- El aislamiento, I(dB), se corresponde con el cociente entre P_1 y P_4, donde P_4 es la potencia que sale por el puerto 4.

$$I(dB) = 10\log\left(\frac{P_1}{P_4}\right) = 10\log\left(\frac{1}{|S_{41}|^2}\right)$$

- La directividad, D(dB), es el cociente entre P_3 y P_4:

$$D(dB) = 10\log\left(\frac{P_3}{P_4}\right) = 10\log\left(\frac{P_1}{P_4}\right) - 10\log\left(\frac{P_1}{P_3}\right) = I(dB) - C(dB)$$

El acoplador formado por dos líneas de transmisión se denomina de sección única y está limitado en ancho de banda. Éste puede ser incrementado utilizando líneas acopladas en serie, tal como se muestra en la Figura 2. Este tipo de acopladores recibe el nombre de multisección.

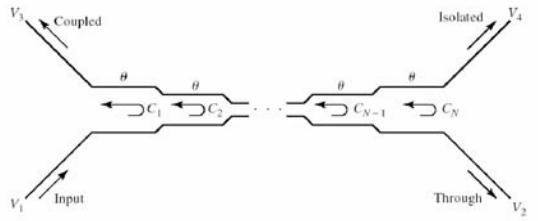


Figura 2. Acoplador direccional de N secciones, basado en líneas acopladas

Generalmente, se diseña el acoplador de manera que resulte simétrico, es decir:

$$c_1 = c_N,\ c_2 = c_{N-1},\ c_3 = c_{N-2},\ \ldots$$

donde *c* representa el factor de acoplamiento de cada sección, siendo *N* impar.

II. DISEÑO

Nuestro circuito acoplador deberá cumplir las siguientes especificaciones de diseño: un número de secciones igual a 3 y 20dB. La frecuencia central será de 3GHz y la impedancia del sistema de 50Ω.

A. *Cálculos previos*

Para una respuesta lo más plana posible para un acoplador de tres secciones (N=3), es necesario que

$$\frac{d^n}{d\theta^n}C(\theta)\Big|_{\theta=\pi/2}=0, \quad para\ n=1,2 \qquad \text{(ec. 1)}$$

donde *C* representa el coeficiente de acoplamiento de tensión,

$$C=\left|\frac{V_3}{V_1}\right|=2\sin\theta\left[C_1\cos 2\theta+\frac{1}{2}C_2\right]=$$

$$=C_1(\sin 3\theta-\sin\theta)+C_2\sin\theta=$$

$$=C_1\sin 3\theta+(C_2-C_1)\sin\theta \qquad \text{(ec. 2)}$$

$$como \qquad \frac{dC}{d\theta}=\left[3C_1\cos 3\theta+(C_2-C_1)\cos\theta\right]\Big|_{\pi/2}=0,$$

$$\frac{d^2C}{d\theta^2}=\left[-9C_1\sin 3\theta-(C_2-C_1)\sin\theta\right]\Big|_{\pi/2}=10C_1-C_2=0 \qquad \text{(ec.3)}$$

y aplicando la condición de que, en la frecuencia central, θ=π/2 y C_0=20dB, se obtiene $C=10^{-20/20}=0.1=C_2-2C_1$ (de ec.2). Sustituyendo en ec.3, resulta

$$C_1=C_3=0{,}0125$$
$$C_2=0{,}125$$

Utilizando las ecuaciones 4 y 5, mostradas a continuación, se obtienen las impedancias características en modo par e impar para cada sección:

$$Z_{0e}=Z_0\sqrt{\frac{1+C}{1-C}} \qquad \text{(ec. 4)}$$

$$Z_{0o}=Z_0\sqrt{\frac{1-C}{1+C}} \qquad \text{(ec. 5)}$$

$$Z_{0e}^1=Z_{0e}^3=50\sqrt{\frac{1{,}0125}{0{,}9875}}=50{,}63\Omega$$

$$Z_{0o}^1=Z_{0o}^3=50\sqrt{\frac{0{,}9875}{1{,}0125}}=49{,}38\Omega$$

$$Z_{0e}^2=50\sqrt{\frac{1{,}125}{0{,}875}}=56{,}69\Omega$$

$$Z_{0o}^2=50\sqrt{\frac{0{,}875}{1{,}125}}=44{,}10\Omega$$

B. *Asignación y obtención de parámetros*

Una vez tenemos los valores necesarios calculados, estamos en disposición de poder emplear la herramienta *TRL synthesis* que nos ofrece ANSOFT DESIGNER (Figura 3), para obtener con ella las proporciones físicas que debería tener cada una de las secciones de nuestro acoplador, para poder ser implementado posteriormente.

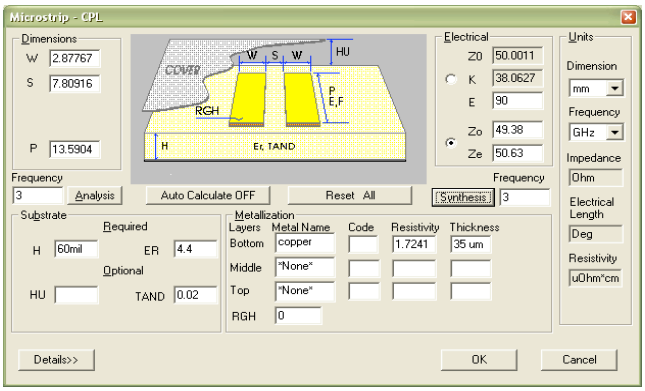


Figura 3. Herramienta TRL aplicada a líneas acopladas (*microstrip*)

Los parámetros que debemos introducir en la herramienta son la impedancia del sistema (Z_0=50Ω), la longitud eléctrica (E=90º), Z_{0o} y Z_{0e} (calculadas anteriormente) y la frecuencia central (F=3GHz).

Con estos datos el TRL *synthesis* nos devolverá la anchura de la línea, la separación entre líneas y la longitud de la línea. Con estos valores ya podemos hacer el esquema del circuito para posteriormente poder ser simulado.

C. *Esquema del circuito y simulaciones*

A la hora de realizar el esquema del circuito, se optó por hacerlo en diferentes diseños de esquemas (a pesar de que en este artículo sólo se refleja la opción más óptima) con el fin de poder mostrar la importancia del acabado del *layout*. En este proceso pudimos ver cómo, para obtener la respuesta que deseamos en nuestro acoplador, fue determinante el tipo de diseño elegido. Esto se puede extrapolar a todos los diseños de circuitos implementados en PCB, sin importar su aplicación.

El primer esquema que se hizo fue el correspondiente a un diseño ideal, donde únicamente se tienen en cuenta las características de las diferentes secciones, considerando despreciable el efecto de la unión entre ellas. El resultado de las simulaciones fue el esperado, pero este diseño no resulta implementable a nivel físico.

Para resolver el problema físico que se presenta con el diseño ideal y que se nos produce en el layout, es necesario hacer un cambio de dirección de la línea. Para ello y en este caso, optamos por introducir un ángulo de 90º en inglete, el cual nos proporcionó la posibilidad de unir las distintas secciones entre ellas.

Otro esquema que se estudió fue el de un diseño con ángulo de 90º curvo. Con este tipo de ángulos, evitamos las aristas de las esquinas, causantes la mayoría de veces de comportamientos eléctricos no deseados en las PCB, trabajando en RF.

Como siguiente caso de diseño, se implementó el circuito con ángulos de 45º curvos. Con este tipo de ángulos, se observó una mejoría en los parámetros S.

Como último esquema de configuración, se implementó el circuito con ángulos de 45º no radiales. En las Figuras 4 a 7 se muestran el esquema, el layout correspondiente, la respuesta del acoplador y la representación 3D.

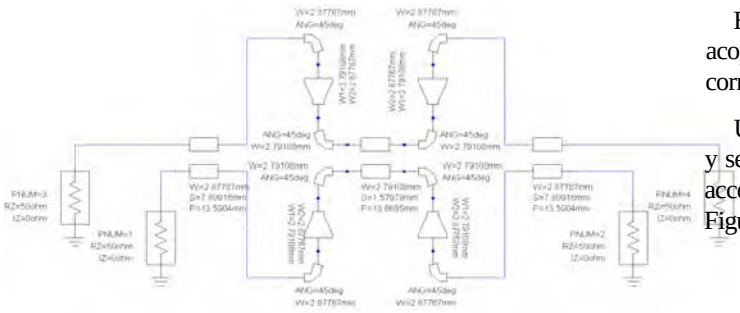


Figura 4. Esquema con codo de 45º

Figura 5. *Layout* con codo de 45º

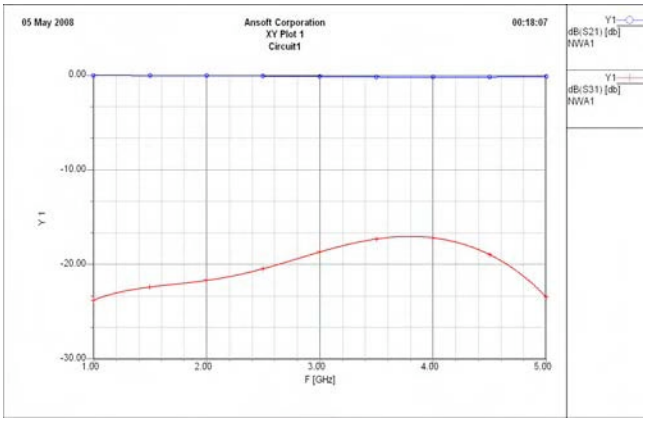


Figura 6. Parámetros S con codo de 45º

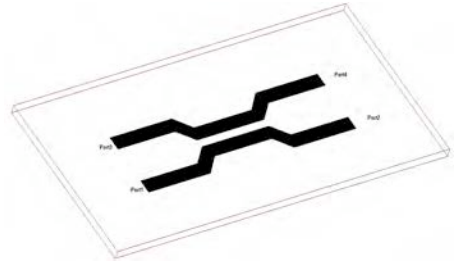


Figura 7. Representación 3D del *layout* con codo de 45º

Dado que la respuesta de acoplamiento entre el puerto de entrada y el directo (S_{21}) es la que más se aproxima al modelo ideal y que la transferencia al puerto acoplado es la que más se acerca a respuesta plana, se ha decidido implementar esta última configuración en PCB.

III. Implementación y Medidas

A partir del *layout* obtenido para el diseño con codos de 45º, se genera el fotolito necesario para la fabricación de la placa de circuito impreso (PCB) que implementará el acoplador direccional de tres secciones.

Se utiliza el procesado químico para su fabricación sobre una placa de doble capa positiva con dieléctrico FR4 de 60mil de grosor y 35µm de cobre, equivalente a la definida en los parámetros de diseño.

En la capa superior aparecen las pistas correspondientes al acoplador. La capa inferior implementa el plano de tierra correspondiente, necesario para la configuración de líneas acopladas.

Una vez fabricada la placa, se corta a las dimensiones apropiadas y se sueldan en cada puerto los conectores de tipo SMA que darán acceso a la inyección de señal y medida correspondientes. Las Figuras 8 y 9 presentan la implementación final del diseño.

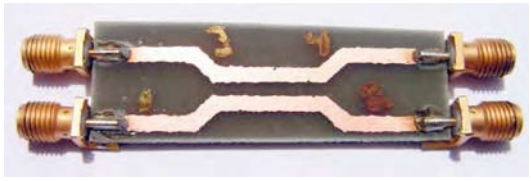


Figura 8. Capa superior, pistas

Figura 9. Capa inferior, plano de tierra

Para llevar a cabo las medidas experimentales, se utiliza el generador de señal de RF Agilent E4433B para inyectar la señal en el puerto de entrada y el analizador de espectro Agilent E4407B para medir la potencia transmitida a cada uno de los puertos directo, acoplado y aislado. Ambos equipos tienen una impedancia de 50Ω, coincidente con la impedancia característica del sistema. Los puertos que no intervienen en la inyección y medida deben estar terminados a la misma impedancia característica para la frecuencia de trabajo.

La señal inyectada se configura con las siguientes características: frecuencia central 3GHz y potencia 0dBm. Esto implica que la medida de potencia transmitida dada por el analizador corresponde a la relación de transmisión de potencia entre el puerto de inyección y el medido.

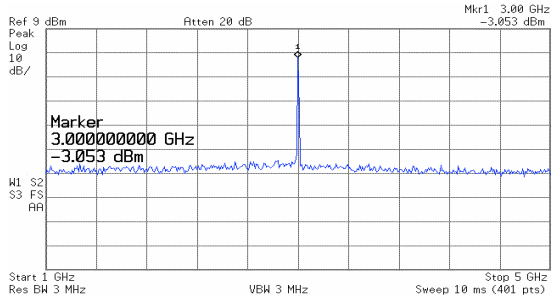


Figura 10. Potencia transmitida al puerto directo 2, inyectando 0dBm – 3GHz en el puerto de entrada 1

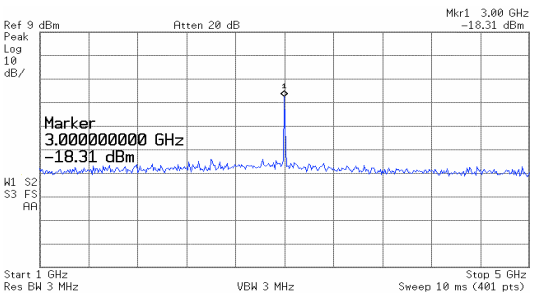


Figura 11. Potencia transmitida al puerto acoplado 3, inyectando 0dBm – 3GHz en el puerto de entrada 1

Así, inyectando señal en el puerto de entrada 1, se obtiene una potencia transmitida en el puerto directo 2 de -3.05dBm, tal como se muestra en Figura 10. La caída de potencia medida, que contrasta con los aproximadamente 0dB dados en simulación, se debe a los efectos de los cables de conexión, los defectos de fabricación de la placa y al desacoplo de impedancias en los puertos no medidos, debidos al uso de terminadores de 50Ω no ideales.

En la Figura 11 aparece la relación de potencia transmitida al puerto acoplado (puerto 3), aproximada a los -20dB dados por condiciones de diseño.

La Figura 12 muestra la relación de potencia transmitida al puerto aislado (puerto 4). El factor de aislamiento no resulta ideal, por los efectos comentados, que producen una desadaptación entre puertos. Sin embargo, puede considerarse aceptable, ya que se haya 7.5dB por debajo del puerto acoplado.

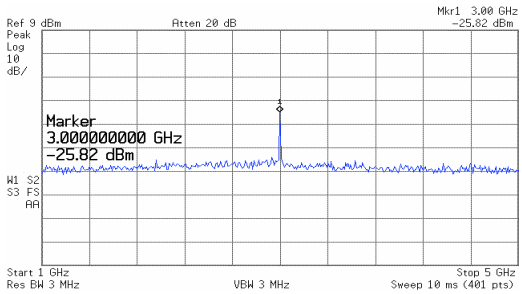


Figura 12. Potencia transmitida al puerto aislado 4, inyectando 0dBm – 3GHz en el puerto de entrada 1

Repitiendo las medidas utilizando los diferentes puertos como puertos de entrada y midiendo la potencia transmitida a los puertos correspondientes, se confirma la característica de reciprocidad definida para el circuito acoplador direccional con líneas acopladas. Esto queda reflejado en los resultados presentes en la Tabla I:

TABLA I
CARACTERÍSTICA DE RECIPROCIDAD

Puerto Entrada	Puerto Directo	Puerto Acoplado	Puerto Aislado
P1: 0 dBm	P2: -3.05 dBm	P3: -18.31 dBm	P4: -25.82 dBm
P2: 0 dBm	P1: -2.72 dBm	P4: -19.13 dBm	P3: -25.06 dBm
P3: 0 dBm	P4: -2.60 dBm	P1: -19.31 dBm	P2: -25.16 dBm
P4: 0 dBm	P3: -2.48 dBm	P2: -19.20 dBm	P1: -25.74 dBm

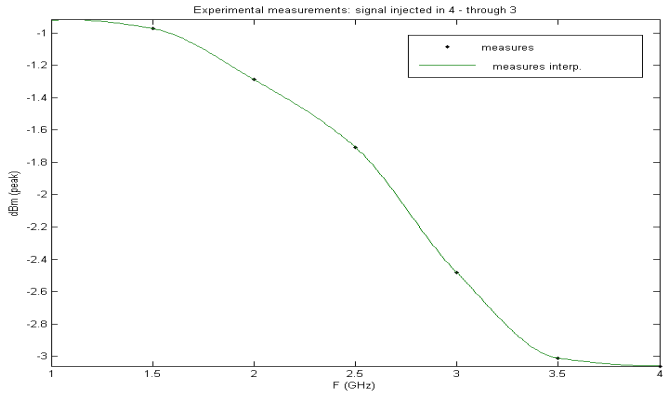


Figura 13. Barrido frecuencial, potencia transmitida al puerto directo 3, inyectando en 4. Medidas experimentales

Utilizando la configuración de P4 como puerto de entrada, se realiza un barrido de frecuencia para la señal inyectada, observando la potencia transmitida al puerto directo (Figura 13). En la Figura 14 se muestran los resultados del mismo barrido por simulación.

Como puede observarse, la tendencia en las medidas experimentales se aproxima a la tendencia dada por simulación. El 'desplazamiento' en la potencia transmitida se debe a los factores ya expuestos, que aparecen al implementar físicamente el circuito y al aplicar la configuración experimental para realizar las medidas.

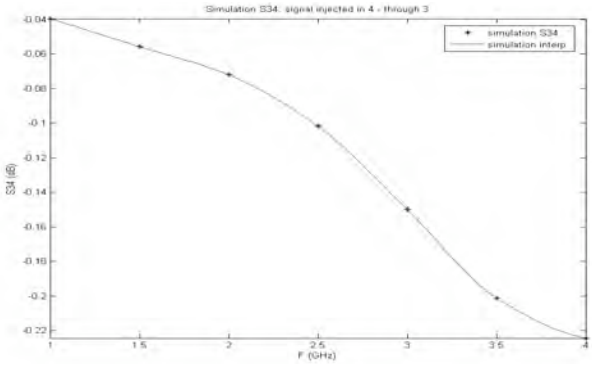


Figura 14. Barrido frecuencial, potencia transmitida al puerto directo 3, inyectando en 4. Resultados de simulación

IV. CONCLUSIONES

Se ha diseñado, simulado, fabricado y medido experimentalmente un circuito acoplador direccional de tres secciones basado en líneas acopladas. Los resultados obtenidos experimentalmente para las potencias transmitidas a los diferentes puertos son coherentes con los resultados de simulación, teniendo en cuenta todos los efectos parásitos que aparecen al implementar físicamente el circuito.

Se ha demostrado mediante medidas experimentales el carácter recíproco del circuito, utilizando los diferentes puertos para inyección de señal.

REFERENCIAS

[1] Tutorial. Overview of Ansoft designer GUI. [Online]. http://omaha.uib.es./A/MasterEE/CAF/material/labo/Ansoft_Designer_PR1.pdf

[2] David M. Pozar, Microwave Engineering, 2nd ed., John Wiley & Sons, Inc., ISBN 0-471-17096-8.

Assignatura impartida per: Dr. Bartomeu Alorda i Dr. Eugeni Garcia

V. CURRICULUM VITAE

Alejandro García Coll, 16 de Marzo de 1.975. Ingeniero. Técnico de Telecomunicaciones Esp. Telemática por la UIB en Marzo del 2.003. Ingeniero Técnico Industrial Esp. Electrónica Industrial por la UIB en Mayo del 2.008. Actualmente alumno del Master en Ingeniería Electrónica (UIB-UPC). Técnico Gestor del Centro de Operación y Despacho de Generación (GESA)

Ivan de Paúl Bernal, Ingeniero Técnico de Telecomunicación por la UIB en 1.999. Actualmente estudiante de Máster en Ingeniería Electrónica (UIB-UPC). Desde marzo de 2003, técnico de investigación en el Grupo de Tecnología Electrónica de la UIB.

Robot camarero

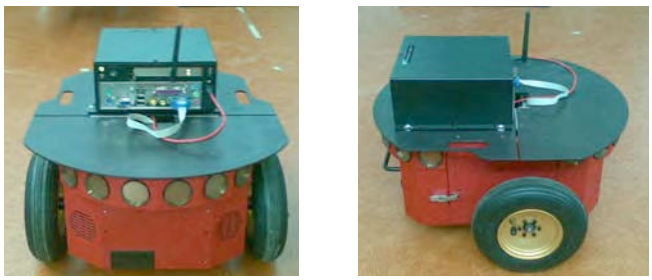
Mª del Mar Sastre Escudero, Ernesto Sigg Rodríguez, Mauro Zanlongo

Asignatura: Robótica

Abstract— **Este documento presenta todos los aspectos utilizados para la implementación de un robot camarero sobre el robot Pioneer de la asignatura Robótica perteneciente a los estudios de ingeniería en informática. Se explicará la arquitectura de control utilizada a través de los niveles estratégico, táctico y ejecutivo, los comportamientos y cálculos realizados, además de una serie de pruebas con el simulador.**

VI. Introducción

La práctica consiste en programar un robot Pioneer con las librerías Aria para implementar la problemática de un negocio de la hostelería (bar, restaurante…). En este caso, el robot corresponde al camarero y las mesas del entorno son los puntos objetivo. La idea de la problemática consiste en que todas las mesas deben estar servidas en el menor tiempo posible y en un orden establecido. Además, en el entorno, podemos encontrar obstáculos como pueden ser columnas o las mismas mesas, las cuales el robot deberá esquivar cuando no sean puntos objetivo, utilizando una técnica de evitación de obstáculos. La realización la práctica se ha ejecutado sobre un robot Pionner 3DX, como muestra las figuras 1 y 2, usando las librerías Aria.



(a) (b)

Fig. 1 (a). Vista frontal del robot Pioneer 3DX. (b). Vista lateral del robot Pioneer 3DX

VII. El Robot Camarero

Como se ha explicado brevemente en la introducción, la implementación simula la gestión de las mesas de un local de hostelería (bar, restaurante…). Para ello, se ha creado un entorno que consta de mesas (objetivos), obstáculos (pueden ser columnas del local o las mismas mesas cuando no son objetivos) y un punto inicial (simula la barra del local). Cada mesa se puede encontrar en uno de los siguientes estados:

1) Vacía: No hay ninguna persona en la mesa.

2) Sin nota: La mesa está ocupada pero todavía no se ha realizado el pedido.

3) Con nota: La mesa está ocupada y ya se ha realizado el pedido.

4) Servida: El robot ha servido la mesa con el pedido solicitado.

Al inicio, el robot se encarga de calcular una prioridad para cada mesa en función de la distancia respecto a la barra (esta prioridad será fija) para, posteriormente, calcular los caminos que deberá seguir para servir y realizar los pedidos de las mesas (por prioridad). De esta manera, el robot realizará dos recorridos: un recorrido de ida para servir a las mesas que ya hayan pedido, y un recorrido de vuelta para tomar nota a las que todavía no lo han hecho. Una vez hecho el camino de vuelta, el robot regresará a la barra. Además, cada vez que el robot llega a la barra, puede percibir cambios en el estado de las mesas, como por ejemplo, que las mesas que ya se hayan servido se pueden vaciar, las mesas vacías pueden llenarse… Estos cambios se producen mediante variables aleatorias. Se ha optado por hacer esta implementación ya que dotar al robot de la capacidad de percibir cambios en el entorno en cualquier momento, significaría trabajar con procesos concurrentes, lo cual no es el objetivo de esta práctica.

En resumen, podemos decir que el algoritmo que sigue el robot al más nivel es el siguiente:

```
Calcular_prioridad_mesas;
Mientras (1) {
        Recorrer_camino_ida (servir mesas);
        Recorrer_camino_vuelta (tomar nota);
        Regresar_a_barra;
        Generar_cambios_entorno (cambios de estado);
}
```

Para finalizar, se mostrará el aspecto que debería tener la práctica simulando un entorno real de un local de hostelería, como podría ser un bar. En la figura 2 se muestra la barra (donde inicialmente empezará el robot a moverse) y 6 mesas a su alrededor que deberá servir a medida que se le envíen peticiones. Además se encuentran 4 columnas en el local que deberá evitar el robot a medida que vaya atendiendo a las mesas.

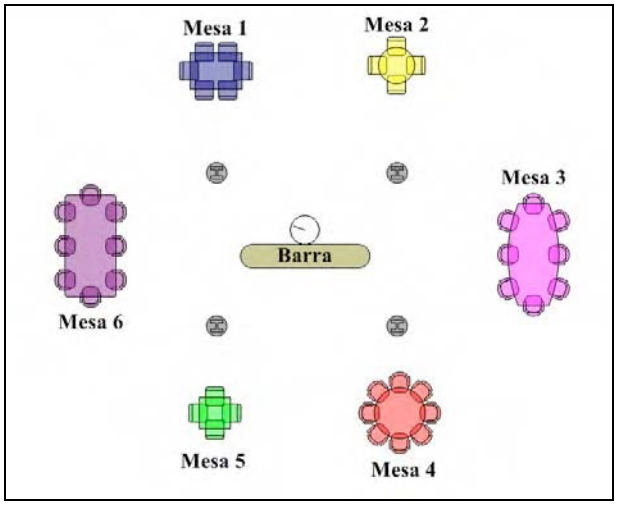


Fig. 2 Entorno real del robot camarero

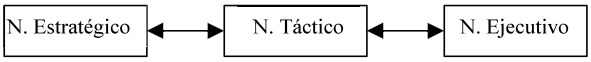


Fig. 2 Arquitectura de control

VIII. Arquitectura del Robot

La arquitectura de control consiste en 3 niveles, que permiten separar la implementación de los programas de alto nivel de la estructura básica del robot. Estos tres niveles son: estratégico, táctico y ejecutivo.

A. *Nivel Estratégico*

Es el nivel donde se ejecuta la misión a alto nivel. Indica al nivel táctico cuáles son los sub-objetivos por donde tiene que pasar el robot para ejecutar la tarea. Este nivel se ejecuta cada cierto tiempo y, a través del nivel táctico, se comprueba si el robot ha llegado al objetivo o no.

B. *Nivel Táctico*

En este nivel se ejecutan los comportamientos que lleva a término el robot, como ir a objetivo, evitar obstáculos… Se encarga de que el robot llegue a los sub-objetivos indicados por el nivel estratégico de manera que no colisione con los obstáculos encontrados en su camino. Este nivel se llama periódicamente por el nivel estratégico.

C. *Nivel Ejecutivo*

Es el encargado de interactuar con la parte física del robot (ultrasonidos y motores). La mayor parte del proceso ejecutivo viene implementado por las librerías del Aria y por el µ-microcontrolador del propio robot. En él se definen los métodos para mover y parar el robot, obtener su posición y los valores de los sensores.

IX. Comportamientos

Para implementar el nivel táctico se utilizan comportamientos reactivos. Un comportamiento es un módulo que a partir de una serie de estímulos exteriores da una determinada respuesta. Esta respuesta es de tipo reactivo, es decir, no se realiza ningún tipo de planificación y sólo depende de los datos que se tengan en ese momento. En este caso, se han generado dos comportamientos: ir a objetivo y evitar obstáculos.

A. *Ir a objetivo*

El comportamiento de ir a objetivo tiene como salida la dirección a la que tiene que ir el robot para llegar a ese punto sin tener en cuenta si existen obstáculos en su camino.

A. *Evitar obstáculos*

El comportamiento para evitar obstáculos tiene como salida la dirección a la que tiene que ir el robot para evitarlos.

La técnica utilizada para implementar dichos comportamientos es la denominada “campos de potencial”. El campo de potencial es un método que consiste en modelizar al robot como si fuera una partícula sometida a una serie de fuerzas. Por ejemplo, en el caso de evitar obstáculos, éstos son los que generan fuerzas de repulsión sobre el robot, mientras que el punto objetivo genera una fuerza de atracción. Cada comportamiento genera una fuerza y la dirección final del robot se obtiene a partir de la suma ponderada del resultado de cada uno de estos comportamientos.

X. Cálculos

Los cálculos empleados para implementar los dos comportamientos comentados en apartados anteriores son los siguientes:

B. *Ir a objetivo*

El funcionamiento de este comportamiento es el siguiente: dados dos puntos, el punto actual (PA) y el punto objetivo (PO) se tiene que definir el vector entre dichos puntos a través de las siguientes fórmulas (1), (2) y (3) a partir de la figura 4.

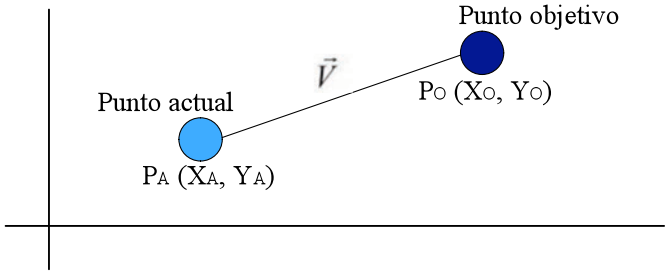


Fig. 4 Ir a objetivo

$$\vec{V} = (X_O - X_A, Y_O - Y_A) \quad (1)$$

$$\vec{V} = \sqrt{X_A^2 + X_O^2} \quad (2)$$

$$\alpha = \arctan\left(\frac{y_v}{x_v}\right) \quad (3)$$

Evitar obstáculos

El funcionamiento de este comportamiento se basa en la técnica “campos de potencial” explicada en el apartado anterior. El robot posee 16 sensores, 8 en la parte delantera y 8 en la parte trasera. Cada sensor genera un vector de repulsión a partir de los datos de cada sensor, es decir, genera 16 vectores de repulsión. A cada vector generado, se le aplica un peso según la posición de cada sensor. En nuestro caso, los sensores traseros tienen 100 veces menos peso que los sensores delanteros. Una vez ponderados se suman para obtener el vector de repulsión total, utilizando la fórmula (4). Cuando se detecte algún obstáculo muy cerca del robot, a una distancia menor a un valor determinado denominado *distancia mínima* (ver punto V.C), se supondrá que el peligro de colisión es inminente. En este caso, se deshabilitan todos los demás comportamientos y sólo estará activo el de evitar obstáculos.

En el caso del robot camarero se han considerado dos posturas en lo que se refiere a la velocidad que debe alcanzar el robot para moverse por el entorno. Por una parte, una versión de la práctica se ha realizado utilizando la velocidad máxima del robot, definida en el fichero de configuración. La segunda versión, la velocidad alcanzada está en función de la cantidad de obstáculos que detecta. Cuando el módulo del vector del comportamiento de evitar obstáculos sea muy alto, la velocidad se reduce, en cambio, cuando el robot se mueve por un entorno donde no encuentra obstáculos cerca de él, la velocidad aumenta.

$$\vec{V} = \sum_{i=1}^{16} p_i \cdot \vec{V}_i \quad (4)$$

XI. PARÁMETROS DEL ROBOT

Para que el robot pueda moverse de un punto objetivo a otro evitando obstáculos, tiene que leer de un fichero de configuración los parámetros de la arquitectura. Este fichero sigue una estructura en el que cada fila corresponde a un parámetro en concreto conocido por nosotros. Los parámetros definidos para el correcto funcionamiento del robot son los siguientes:

- *maxDist:* Es la distancia máxima a partir de la cual los objetos detectados no se consideran obstáculos, definida en mm.
- *velMax:* Es la velocidad máxima a la que puede moverse el robot hacia los puntos objetivo, definida en mm./s.
- *distMin:* Es la distancia mínima de colisión, definida en mm.
- *distObj:* Es la distancia mínima a partir de la cual consideramos que hemos llegado al objetivo, definida en mm.
- *DRot:* Es la diferencia máxima entre la rotación actual del ángulo y la que tendrá que tener, definida en grados.
- *T:* Es el período de tiempo entre llamadas, definido en ms.
- *pesoObjetivo:* Es el peso del comportamiento de ir hacia el objetivo.
- *pesoObstaculo:* Es el peso del comportamiento de evitar obstáculos.
- *numMesas:* Es el número de mesas que se encuentran en el entorno.
- *Parámetros de las mesas:* A partir de aquí se encuentran los parámetros asociados a cada mesa, a través de un conjunto de 2 filas para cada mesa. La primera fila corresponde al estado inicial de cada mesa, relacionado con los 4 estados mencionados en el apartado II. La segunda fila corresponde a la posición (x, y) de cada una de las mesas. Esta posición no se modifica a lo largo de la ejecución de la práctica.

XII. PRUEBAS EXPERIMENTALES

Una vez explicado todo el funcionamiento del robot camarero, se mostrará a través de la figura 5 la siguiente prueba: ir desde un punto inicial (simulando la barra) a un punto objetivo (simulando una mesa) con un obstáculo situado en medio del camino a realizar. En dicha figura se observan dos caminos: la flecha roja simula el camino que debería hacer el robot si no hubiera ningún obstáculo en su camino para alcanzar el objetivo. Por otro lado, la flecha verde simula el camino que debe realizar el robot en caso de encontrarse un obstáculo en su camino. Como se puede observar, sigue el método utilizado que se ha explicado anteriormente de evitar obstáculo.

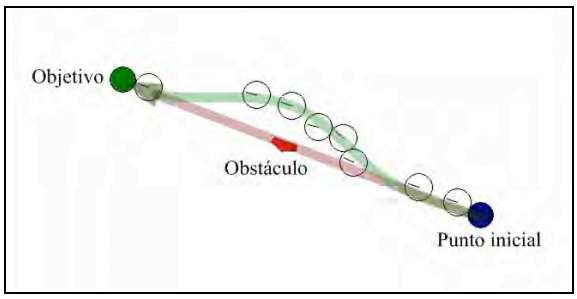


Fig. 5 Técnica evitar obstáculo

A continuación, en la figura 6, se muestra una captura de pantalla de los resultados que se han obtenido en una de las ejecuciones de la práctica a medida que el robot se va moviendo por el entorno realizando sus tareas. En un primer momento se visualizan el estado de cada una de las mesas ordenadas por la distancia que debe recorrer el robot para llegar a ellas. En este caso, la mesa 2 está vacía, la mesa 3 está ocupada pero todavía no ha sido atendida por el robot y las mesas y 0 ya han realizado el pedido al robot. Según se ha explicado anteriormente, el robot realiza dos caminos: uno de ida para servir a las mesas que ya hayan pedido y uno de vuelta para tomar nota a las que todavía no lo han hecho. De esta manera, la mesa 3 esperará a que el robot haya servido antes a las mesas 1 y 0. Una vez realizado los dos caminos regresa a la barra. A partir de ahí, se vuelven a generar cambios en el estado de las mesas, los cuales repercutirán en el camino que deberá recorrer el robot.

Se han realizado una serie de pruebas sobre el robot real (Pioneer 3DX) donde se ha podido observar la correcta ejecución de la práctica. Para ello se han tenido que realizar pequeños cambios en los parámetros del robot, como por ejemplo, *distMin* o *Drot* (ver punto VI). En la figura 7 se muestra un momento de la ejecución de la práctica donde se observa el comportamiento de evitar obstáculos.

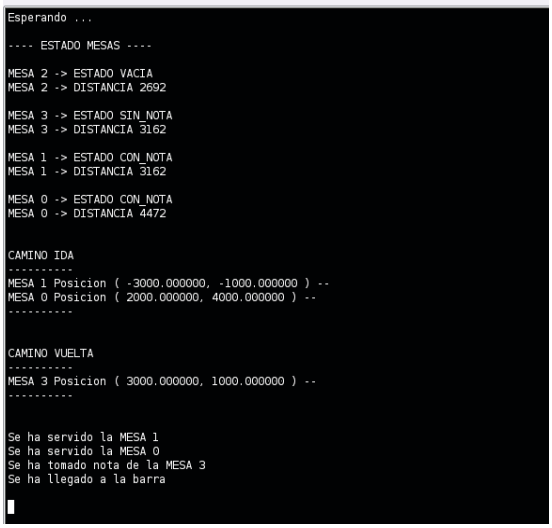


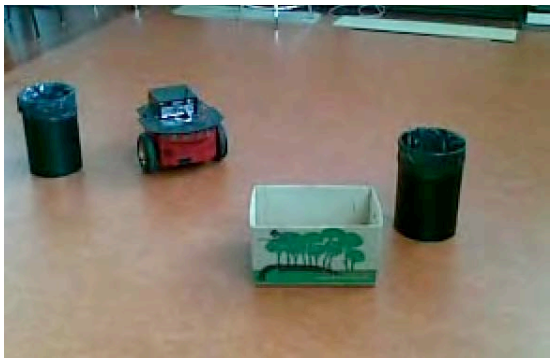
Fig. 6 Debug robot camarero

XIII. CONCLUSIONES Y MEJORAS

A modo de conclusión se han tenido en cuenta una serie de aspectos que pueden servir como punto de partida para posibles mejoras en el robot:

- La posibilidad de tener en cuenta el número de personas que ocupan cada mesa: con este dato se pueden calcular nuevas prioridades y nuevos caminos.
- Se pueden usar técnicas basadas en mapas para mejorar el recorrido del camino y la evitación de obstáculos.
- Existe la posibilidad de dotar al robot de capacidad para percibir cambios en el entorno en cualquier momento del ciclo de ejecución. Una posibilidad sería implementar un proceso concurrente que envíe "nuevos eventos" al robot.

Fig. 7 Prueba real – evitar obstáculos



Mª del Mar Sastre Escudero. Estudiante de 5º de ingeniería informática. Ingeniera Técnica en Informática de sistemas (UIB-2006). Trabajando actualmente en el Centro de Tecnologías de la Información (UIB) en desarrollo del proyecto de mecanización de la FUEIB.

Ernesto Sigg Rodríguez. ernesto.sigg@uib.es . Ingeniero Técnico en Informática de sistemas (UIB-2006). Trabajando actualmente para la Fundació Universitat-Empresa (UIB) en aplicaciones de vigilancia tecnológica.

Mauro Zanlongo. Estudiante de 5º de ingenieríainformática. Ingeniero Técnico en Informática de sistemas (UIB-2006). Trabajando actualmente para Axis Data S.L. en el desarrollo de aplicaciones B2B. Otros articulos: "The Intelligent Butler: a Virtual Agent Prototype for Disabled People Assistance".

Medición de la Respuesta Frecuencial de un Filtro Pasa-Banda mediante LabVIEW

Efrén Benítez Villanueva, Eloi Pardo Gómez del Cerro, Miguel Torres Oliver

Asignatura: Instrumentación Electrónica II

XIV. INTRODUCCIÓN

El objetivo de la práctica es desarrollar un analizador de redes virtual que permita medir la respuesta frecuencial de un sistema lineal en régimen permanente sinusoidal.

Existen dos formas de poder realizar la medida de la respuesta frecuencial. El primero y más simple consiste en excitar al sistema con una onda sinusoidal y observar la salida. Realizando un barrido en frecuencia y se mide la amplitud de la salida y el desfase de la señal respecto a la entrada, obteniéndose la respuesta frecuencial del sistema (Vo/Vi).

La segunda opción es medir en baja frecuencia, excitando el sistema con una fuente de señal de ruido blanco, para este caso la función de transferencia se realiza a partir de las densidades espectrales de potencia en la entrada y en la salida.

El método elegido fue el primero, excitando el sistema con onda sinusoidal y observar como varía la salida en función de la entrada.

Para llevar a cabo el estudio de la herramienta virtual se utilizaron estos dispositivos: fuente de alimentación, generador de funciones, osciloscopio digital y multímetro digital. Estos instrumentos son controlables mediante bus GPIB, nos ayudaron a automatizar las medidas y desarrollar un VI en LabVIEW, haciendo la misma función que un analizador de redes real. [2]

El filtro implementado para el estudio fue un filtro activo de segundo orden pasa-banda. Formado por un amplificador operacional modelo tl071, dos condensadores y tres resistencias.

XV. CÁLCULOS PREVIOS

C. Diseño del Filtro

Como se ha dicho anteriormente el filtro utilizado es un filtro pasa-banda activo de segundo orden (Fig 1). El filtro es activo debido a que en el circuito tenemos un amplificador operacional.

Se eligió este tipo de filtro debido a las ventajas que tiene sobre los filtros pasivos como pueden ser: ajuste de la ganancia, mejoras de los efectos de cargas (Zin elevada y Zout baja), bajo coste del amplificador operacional, flexibilidad para poder realizar la función de transferencia deseada y la integrabilidad.

Aunque no nos olvidamos de las desventajas que presentan estos tipos de filtros, como son: ancho de banda finito de los componentes activos, no idealidades de los amplificadores operacionales, necesidad de alimentación, distorsión de los elementos activos y ruido.

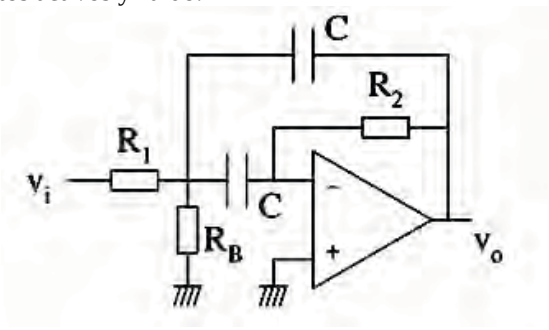


Fig. 1 Esquema básico del filtro pasa-banda activo de segundo orden

D. Cálculos Teóricos

[1]Para realizar los cálculos nos basamos en la función de transferencia representativa de este tipo de filtros, ecuaciones (1), (2) y (3),

$$H = \frac{K(w_o/Q)s}{s^2 + (w_o/Q)s + w_o^2} \qquad (1)$$

donde

K: ganancia para ω = ωo.

ωo: frecuencia de pulsación o frecuencia angular central.

$$Q = \frac{w_o}{w_N - w_L} = \frac{w_o}{2\pi BW} = \frac{f_o}{BW} \quad (2) \qquad w_o = \sqrt{w_H \cdot w_L} \quad (3)$$

Realizando los cálculos oportunos sobre el circuito, obtenemos que la siguiente ecuación (4),

$$H = \frac{(-1/R1 \cdot C)}{s^2 + (2/R2 \cdot C)s + (R1 + RB/R1 \cdot R2 \cdot RB \cdot C^2)} \qquad (4)$$

Si imponemos el valor de las resistencias R2=2R1 y R1=7RB (R2=14RB) obtenemos un valor de ganancia K=-1. El valor de frecuencia de resonancia (fres) ωo (5) y el factor de calidad del filtro (6),

$$w_o = \sqrt{\frac{R1 + RB}{R1 \cdot R2 \cdot RB \cdot C^2}} \quad (5) \qquad Q = \frac{R2 \cdot C \cdot w_o}{2} \quad (6)$$

Por último imponemos valores conocidos a las resistencias RB=4,7KΩ, R1=33KΩ y R2=66KΩ y los condensadores, que para este caso los dos son iguales, por tanto C=100nF.

Como resultado tenemos finalmente un factor de calidad Q=2, una frecuencia de resonancia ωo≈100Hz y un ancho de banda BW≈50Hz. En la figura 2 se pueden ver los parámetros de una función de transferencia tipo.

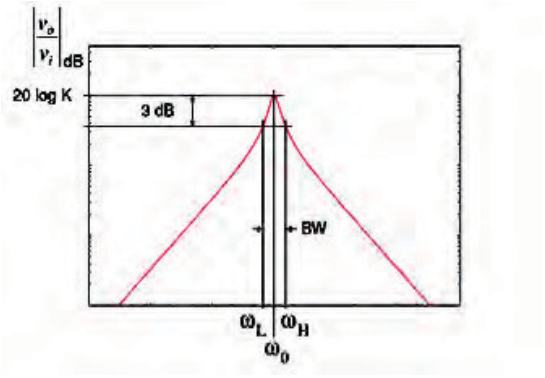


Fig 2. Función de Transferencia típica de un filtro Pasa-Banda activo. [2]

XVI. DESCRIPCIÓN DEL VI DESARROLLADO

A. *Controles que aparecen en el panel*

- Amplitud: se especificará la amplitud de la señal de entrada del circuito.
- Número de pasos: con este control el usuario podrá seleccionar el número de puntos que desee que se realicen en el barrido de frecuencias.
- Frecuencia Baja: el usuario podrá seleccionar la frecuencia inicial del barrido.
- Frecuencia Alta: seleccionaremos la frecuencia final del barrido de frecuencias.
- Escala: control encargado de indicar el tipo de barrido frecuencial. Los valores que puede tener son dos:
  0 → Escala logarítmica
  1 → Escala lineal

Los valores de los controles deben cumplir una serie de requisitos:
- Número de pasos > 0
- Amplitud > 0
- Frec. inicial < fres < Frec. Final

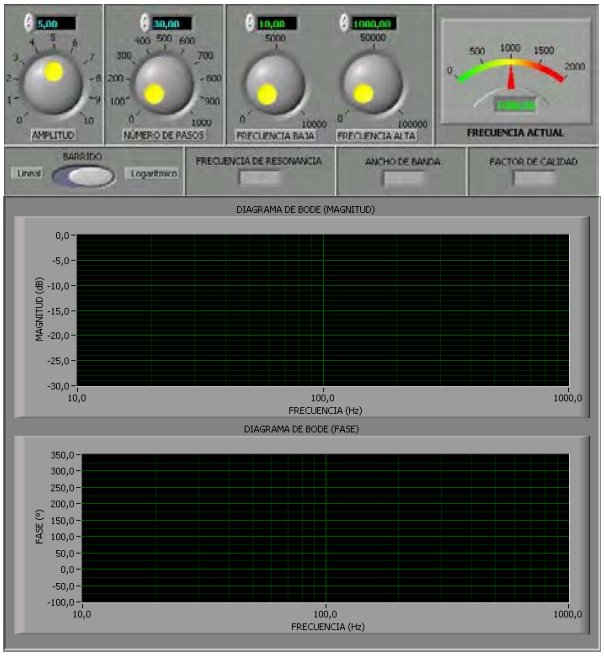


Fig 3.Panel frontal del VI antes de simular

B. *Indicadores que aparecen en el panel*

- Frecuencia de Resonancia: mostramos el valor de la frecuencia de resonancia calculada.
- Ancho de banda: mostramos el resultado del cálculo del ancho de banda de nuestro filtro.
- Factor de calidad: valor calculado del factor de calidad del filtro.
- Diagrama de Bode: con este indicador representamos el módulo de la respuesta frecuencial de nuestro filtro pasa-banda.
- Diagrama de Fase: en este indicador graficamos la Fase.
- Frec. Actual: mostramos la evolución del valor de la frecuencia durante el barrido Fig 3.

C. *Funciones*

Se especificaron una serie de funciones mínimas que se debían implementar:

1) Funciones mínimas:
   a) Gráfica con el Módulo de la respuesta frecuencial.
   b) Posibilidad de poder seleccionar la frecuencia inicial y final del barrido así como el número de puntos de este.
   c) Posibilidad de seleccionar la amplitud de señal de excitación del DUT.

Aparte de estas funciones mínimas, se dieron como opción implementar una serie de funciones adicionales, de las cuales se han realizado las siguientes:

2) Funciones adicionales:
   a) Posibilidad de elegir entre un barrido logarítmico y lineal.
   b) Aparte de la gráfica también tiene de proporcionar el BW (-3dB) del filtre, la frecuencia de resonancia (fres) así como el factor de calidad Q=fres/BW.
   c) Obtener también la fase de la respuesta frecuencial

D. *Descripción del código VI*

Se trata de un circuito secuencial como se puede ver en la fig 4. En el primer bloque inicializamos los instrumentos.

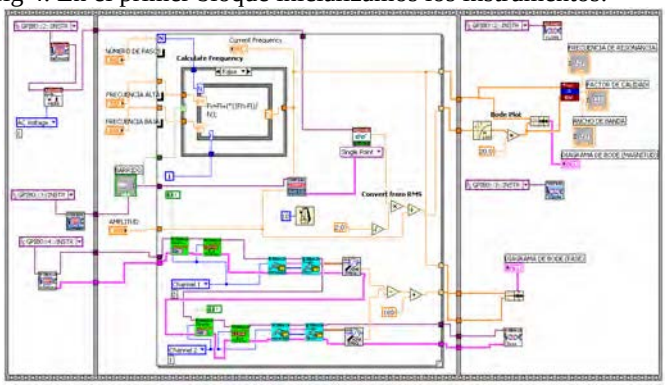


Fig 4.Esquema del VI con sus tres bloques diferenciados

Con los valores obtenidos por los instrumentos y los controles especificados por el usuario procedemos a realizar los cálculos.

El usuario puede decidir si desea un barrido logarítmico o lineal. Se utilizarán formulaciones distintas para calcular la

escala en cada caso. El valor de esta escala irá en función de la frecuencia máxima y mínima y del número de puntos de muestreo y su valor se guardará en un array.

Además realizando lecturas de los instrumentos se calculará el valor de la magnitud y de la fase para esa frecuencia y se guardarán en arrays distintos, como se puede ver en el diagrama de la figura 5.

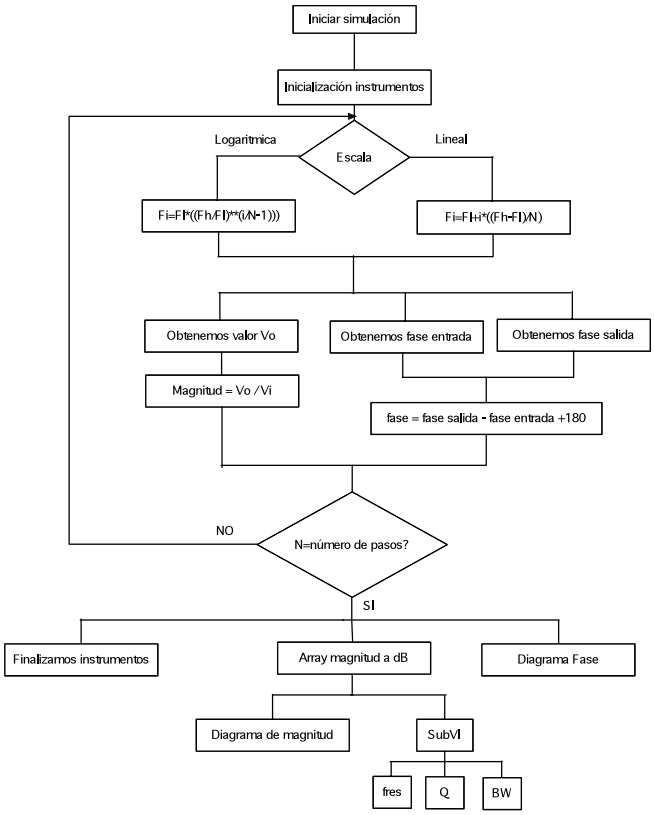


Fig 5.Diagrama de flujo de VI

Todo este proceso del segundo bloque se realizara N veces (numero de pasos que se ha especificado por el usuario). Al finalizar obtendremos tres arrays de N valores. En uno de ellos guardaremos el valor de la frecuencia donde se han realizado los muestreos, los otros dos guardan los valores de la magnitud y de la fase respectivamente que se han obtenido en esos muestreos.

En el último de los bloques se interpretan y muestran por pantalla los resultados previamente obtenidos. El array de la magnitud lo pasamos a dB y como conocemos los puntos donde se han obtenido los valores (primer array) podemos realizar su diagrama.

Con estos dos arrays (magnitud y frecuencias) podemos sacar los valores de la frecuencia de resonancia, el ancho de banda y el factor de calidad, para ello se ha realizado un SubVI, como se puede ver en diagrama de la figura 6.

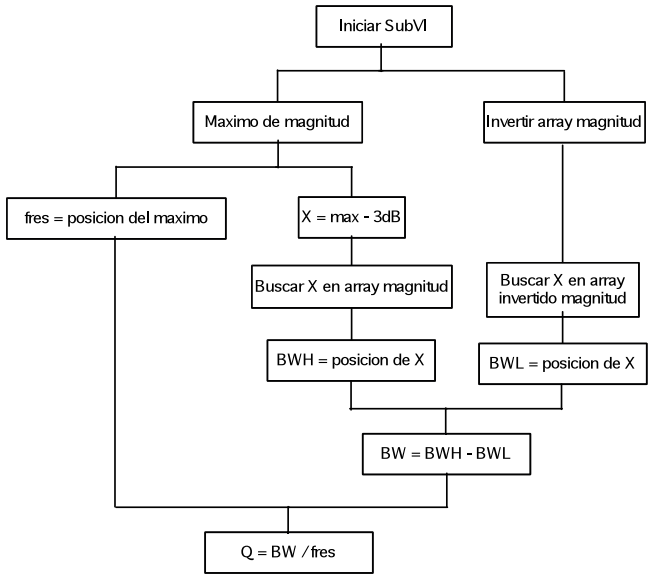


Fig 6.Diagrama de flujo de SubVI

Para calcular la frecuencia de resonancia tenemos que buscar el valor máximo en el array de la magnitud y luego buscar su posición en él.

Para calcular el ancho de banda hay que buscar los valores de la magnitud que están 3dB por debajo del máximo (frecuencia de resonancia) y posteriormente buscar su posición. Para calcular el primer punto pasaremos por el array buscando el valor mayor de la frecuencia de resonancia menos 3 dB (f_L). Como tan solo nos devuelve un valor, el proceso se ha de repetir para encontrar el otro punto que tenemos con el valor de la frecuencia de resonancia menos 3 dB (f_H). Para calcularlo pasaremos por el array de nuevo pero esta vez de manera invertida. Para obtener el ancho de banda (BW) lo que haremos será restar las frecuencias f_H-f_L.

Finalmente el factor de calidad del filtro lo podemos obtener con la división del ancho de banda por la frecuencia de resonancia. En la figura 7 se puede comprobar como se ha ealizado en LabVIEW el sub VI para el cálculo de los parámetros descritos anteriormente.

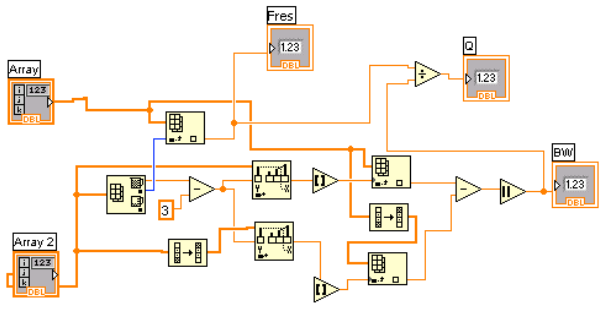


Fig 7.Esquema del SubVI para el cálculo de la frecuencia de resonancia, el ancho de banda y el factor de calidad.

Por último mostramos los tres valores que nos devuelve el SubVI (frecuencia de resonancia, factor de calidad y ancho de banda).También se muestra el diagrama de Bode y la fase

obtenido con el array de los valores de la fase tal y como hemos hecho con el de los valores de la magnitud. Por último cerramos los instrumentos que habíamos inicializado.

XVII. RESULTADOS DEL ESTUDIO

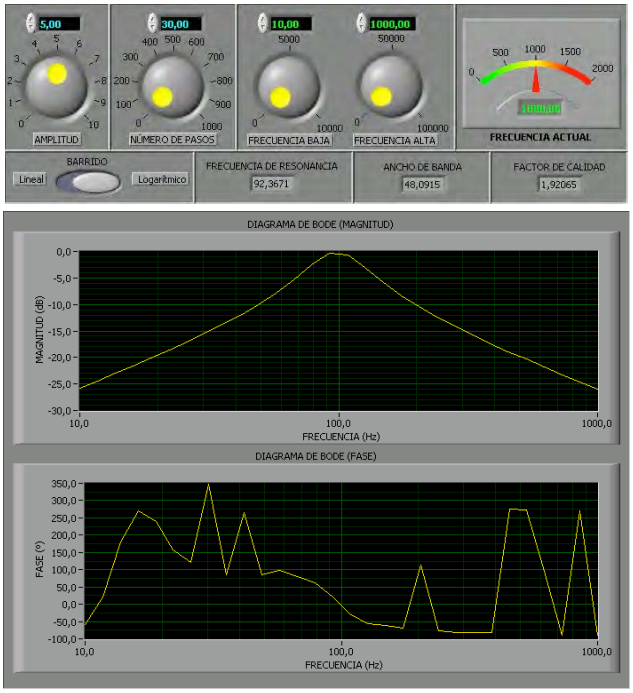


Fig 8.Panel frontal del VI después de obtener los resultados

En la figura 8 podemos ver los resultados obtenidos durante una simulación con una amplitud de 5V en la señal de entrada, con 30 pasos en la medición de los parámetros. Una frecuencia inicial de 10Hz y una frecuencia final de 1MHz, con un barrido logarítmico. Obteniéndose una frecuencia de resonancia de 92Hz aproximadamente, un ancho de banda de unos 48Hz aproximadamente y un factor de calidad de 1,92.

A continuación se hace una comparativa entre los valores obtenidos analíticamente y en LabVIEW. Como se puede ver en la figura 9.

Valores prácticos:	**Valores teóricos:**
ωo = 92,3671 Hz	ωo = 96,7507 Hz
BW = 48.0915Hz	BW = 48.3753Hz
Q = 1.92065	Q = 2

Fig 9.Comparativa de los resultados analíticos y experimentales.

En cuanto al diagrama de la fase hay que decir que lo calcula correctamente en el entorno de la frecuencia de resonancia, ya que empieza en 90º debido al cero en el origen del filtro y que baja aproximadamente 180º en dos décadas debido a los 2 polos que tiene en Fres. Creemos que los picos generados por los puntos más alejados del centro son debidos a que el osciloscopio no es capaz de detectar variaciones de fase con la señal tan atenuada (del orden de -20 dB).

XVIII. CONCLUSIONES

Podemos estar satisfechos con los resultados obtenidos, ya que las diferencias entre los valores obtenidos analíticamente y los valores obtenidos con la simulación son mínimas. Esto se debe a las no idealidades de los componentes activos, tolerancia de las resistencias utilizadas en el montaje del filtro sobre placa proto-board, etc…

Se ha de puntualizar que la calidad de los resultados depende del número de puntos del muestreo elegidos en el panel frontal de controles de LabVIEW. Es decir, cuanta mayor precisión tenga el muestreo mejores resultados obtendremos. Por lo que si el número de puntos es pequeño, tanto las gráficas como los parámetros calculados variarán respecto a los resultados ideales.

Las principales dificultades encontradas en la realización de esta práctica han sido relacionadas con el uso del programa LabVIEW cuyo manejo desconocíamos hasta la realización de las prácticas introductorias.

AGRADECIMIENTOS

Esta asignatura ha sido impartida por los profesores Jaume Verd, Bartomeu Alorda y Vicenç Canals (ETG – Departamento de Física de la UIB).

REFERENCIAS

[3] M.A. Pérez, J.C. Álvarez, J.C. Campo, Fco. J. Ferrero, G.J. Grillo, "Instrumentación Electrónica" Editorial Thomson, ISBN 8497321669.
[4] Ramon Pallàs "Instruments electrònics bàsics", Publicació Universitat Politècnica de Catalunya, 1992, ISBN 8476532326
[5] "Instrumentación electrónica", M. A. Pérez et al., Thomson (2004).
[6] "LabVIEW 7i. Programación gráfica para el control de instrumentación", A.M.Lázaro, Ed. Paraninfo (2005), ISBN: 84-9732-391-2.
[7] Jaume Verd y Bartomeu Alorda , Práctica 1 Instrumentación Electrónica II 2008, Departamento de Física, UIB.

Efrén Benítez Villanueva (Jerez de la Frontera, Cádiz, 1983) és estudiant de tercer curs d'Enginyeria Tècnica Industrial (esp. Electrònica Industrial) a la UIB

Miquel Torres Oliver (Palma de Mallorca, 1987) és estudiant de tercer curs d'Enginyeria Tècnica Industrial (esp. Electrònica Industrial) a la UIB.

Eloi Pardo Gómez del Cerro (Barcelona, 1986) és estudiant de tercer curs d'Enginyeria Tècnica Industrial (esp. Electrònica Industrial) a la UIB.

Streaming de Audio/Video. Protocolo RTSP

David Mateos Costilla, Samuel Reaño Montoro

Serveis Telemàtics

Resumen— Este artículo pretende hacer un resumen de las tecnologías existentes para la difusión de contenidos multimedia a través de internet, centrándose en el protocolo de transmisión RTSP.

I. Introducción

"Streaming" es un vocablo de procedencia inglesa que significa, literalmente, manar. No obstante su acepción más popular hoy en día es seguramente la que hace referencia a la técnica asociada a la reproducción de información audiovisual en tiempo real

Se utiliza para minimizar el tiempo de espera en la transferencia de un archivo multimedia desde una red, normalmente internet. Permite ver y escuchar los archivos mientras se hace la transferencia sin que sea necesaria la descarga completa del archivo.

II. Recorrido histórico

En la década de los '90 al avance imparable de la potencia de cálculo, tal y como predijera en 1965 Gordon E. Moore en su famosa ley [1], se le unió la enorme expansión que experimentó internet en aquella época. En estos años se empezó a barajar la posibilidad de compartir contenidos multimedia en tiempo real a través de la red de redes. El problema era que la información era demasiado pesada y las conexiones de la época demasiado lentas para este tipo de aplicación.

En 1992 el Motion Pictures Experts Group (MPEG) aprueba un algoritmo de compresión de audio llamado a transformar el modelo de distribución musical para siempre. Se trataba del estándar MPEG1 Audio Layer 3, conocido popularmente como MP3 [2], que permitía unas tasas de compresión de, aproximadamente 1:10 en relación al audio digital PCM sin procesar, manteniendo una calidad de sonido suficientemente fidedigna al original.

Esta sensible disminución de la cantidad de información a transmitir y el progresivo aumento en el ancho de banda de las conexiones a internet dio lugar a la aparición de las primeras redes de intercambio de ficheros multimedia mediante protocolos *"Peer To Peer"* o P2P, entre las que podemos destacar por su popularidad, la red Napster. Cuestiones éticas a parte, la popularización masiva del intercambio de contenidos multimedia a través de la red hizo que mucha gente se diera cuenta del potencial de las nuevas telecomunicaciones en combinación con estos nuevos y potentes algoritmos de compresión. La posibilidad de ofrecer video y audio en tiempo real a cualquier punto del planeta a través de internet ya no parecía imposible y gigantes multinacionales como Microsoft, Real o Apple empiezan a desarrollar sus primeros protocolos de streaming.

III. Protocolos de Transporte en Tiempo Real

A. RTP (Real-Time Transport Protocol)

Protocolo de transporte en tiempo real que conduce la entrega de paquetes coordinada por RTSP y RTCP. Se basa en el envío 'poco fiable', es decir de mejor esfuerzo, en tiempo real de datos sobre UDP. Este protocolo se suele usar conjuntamente con RTSP para las tareas de control del flujo de datos mediante sesiones.

En contra de lo que su nombre pueda sugerir, la utilización del protocolo RTP no asegura que la transmisión sea en tiempo real, pero sí que aumenta la sincronización y el control sobre los contenidos en tiempo real (audio y/o video). Es decir, se puede enviar el audio y el video por "flujos" separados y mediante este protocolo, sincronizarlos posteriormente en el destino.

B. RTCP (Real-Time Transport Control Protocol)

Protocolo de control de transporte en tiempo real que se encarga de tareas de comunicación e información para el correcto control del flujo de datos de RTP.

Los paquetes de este protocolo no transportan datos multimedia, sino que trabaja con RTP en el transporte y empaquetado de los datos. Se usa para transmitir los parámetros de una sesión multimedia, y su función principal es informar a cerca de la calidad del servicio (QoS). Durante una sesión, RTCP transmite periódicamente paquetes de control a todos los participantes de la misma para controlar el estado de la conexión en todo momento.

C. RTSP (Real Time Streaming Protocol)[3]

Protocolo de flujo de datos en tiempo real no orientado a conexión que se utiliza para definir cómo se hará el envío de información entre el cliente y el servidor.

Este protocolo trabaja a nivel de aplicación y controla que la entrega de datos se realice correctamente, pues el tipo de contenido con el que se trabaja normalmente al hacer streaming es muy sensible a la sincronía temporal (o a la falta de ella). Así pues se podría considerar que el RTSP actúa como si de una especie de mando a distancia de red para servidores multimedia se tratase.

RTSP define diferentes tipos de conexión y diferentes conjuntos de requisitos, para intentar conseguir siempre un envío de flujo de datos a través de redes IP lo más eficiente posible. Además establece y controla uno o más flujos sincronizados de datos como audio y video. A tal fin se definió el uso de sesiones, mediante identificador único, en este protocolo.

Es independiente del protocolo de transporte y puede funcionar tanto sobre UDP, RDP o TCP. Durante una sesión, un cliente puede abrir y cerrar conexiones fiables de transporte con el servidor mediante peticiones RTSP.

Los flujos controlados por RTSP pueden usar RTP como hemos visto, pero el modo de operación de RTSP es independiente del mecanismo de transporte usado para

transmitir el continuo flujo de datos. Sin embargo, en la mayoría de casos se utiliza TCP para el control del reproductor y UDP para la transmisión de datos con RTP.

D. Propiedades del protocolo

- Extensible: nuevos métodos y parámetros pueden ser añadidos a RTSP fácilmente.
- Seguro: RTSP usa los mecanismos de seguridad de la web, cualquiera a nivel de transporte. Se pueden aplicar directamente los mecanismos de autentificación de http.
- Capacidad multi-servidor: Cada flujo de contenido perteneciente a una misma presentación puede residir en diferentes servidores.
- Separación del control del stream y la iniciación de la conferencia: El control de stream no tiene nada que ver con las invitaciones de conferencia a un servidor. En particular para estos casos se suele usar SIP.
- Neutral respecto de la descripción de las presentaciones: no impone ninguna descripción particular o formato concreto de metafile.
- Muy similar a http: cuando es posible RTSP reutiliza los conceptos de http, por lo cual puede usar la infraestructura ya establecida.
- Capacidad de negociación: si las características básicas están desactivadas, hay un mecanismo para que el cliente pueda determinar qué métodos van a ser implementados. Esto permite a los clientes usar la interfaz más apropiada.

E. Operaciones soportadas

- Proporcionar contenidos multimedia desde un servidor dedicado: El cliente puede pedir una descripción de la presentación vía HTTP o cualquier otro método. Si la presentación está en multicast [4], la descripción de la presentación contiene la dirección de multicast y los puertos que se deben usar para la recepción de media. Si por el contrario la presentación es enviada solamente de forma unicast[5], el cliente proporcionará el destino por razones de seguridad.
- Invitación de un servidor de streaming a una conferencia: Un servidor de streaming puede ser invitado a unirse a una conferencia ya existente, para reproducirla en una presentación. Este modo es muy útil para la distribución de aplicaciones educacionales.
- Añadido de contenido multimedia a una presentación existente: Particularmente para presentaciones en vivo, es muy útil si el servidor puede avisar al cliente sobre los nuevos contenidos adicionales disponibles.

F. Peticiones RTSP

- OPTIONS: Esta petición puede ser enviada en cualquier momento y no influye en el estado de la sesión. Es utilizada en el inicio, para establecer la sesión con los parámetros necesarios, como el número que se nos asigna y cierta información del servidor como la versión o los métodos soportados. También se utiliza, por ejemplo, cuando un usuario pretende realizar una petición no estándar.
- DESCRIBE: Se encarga de inicializar la sesión. Obtiene una descripción del objeto requerido, principalmente el flujo necesario para la correcta reproducción.
- SETUP: Especificará los parámetros de transporte tales como el protocolo a utilizar, el puerto de recepción o el puerto para los paquetes RTCP.
- PLAY: Indica al servidor cuando debe comenzar el envío de datos, con los parámetros especificados en el SETUP. El cliente nunca deberá enviar una petición PLAY sin antes haber recibido una confirmación SETUP.
- PAUSE: Detiene temporalmente el flujo de datos.
- TEARDOWN: Petición para detener completamente el flujo de datos y liberar los recursos asociados.
- GET_PARAMETER: Como su propio nombre indica, sirve para recuperar el valor de algún parámetro de la presentación en curso.
- SET_PARAMETER: Análogamente a la anterior petición, sirve para establecer el valor de un parámetro de la presentación en curso.
- REDIRECT: Petición que informa al cliente de que debe conectar a otra dirección de servidor.

En una sesión RTSP común los intercambios que se producen son los siguientes: el cliente envía una petición RTSP, con formato HTTP al servidor del streaming. Cuando se establece la conexión con el mismo, normalmente vía TCP, se manda una petición OPTIONS pidiendo los datos al servidor para configurar una sesión. Una vez hecho esto, el cliente está preparado para pedir mediante un DESCRIBE una descripción de la presentación concreta a reproducir, a la cual el servidor responde con todos los valores de inicialización necesarios para dicha presentación. La siguiente petición será un SETUP para cada flujo de datos que se quiera reproducir, obteniendo los protocolos aceptados para el transporte. Con todos los datos conocidos ya podemos pedir mediante un PLAY que comience el envío de los datos por parte del servidor.

Durante la sesión el servidor y el cliente intercambian periódicamente mensajes de PING mediante SET_PARAMETER. Al detener o terminar el flujo de datos el cliente envía las estadísticas de la sesión también con SET_PARAMETER. Habiendo concluido el streaming el cliente envía un TEARDOWN para liberar definitivamente los recursos.

TABLA I

PETICIONES RTSP. DIRECCIÓN DEL TRÁFICO

Petición	**Dirección**
DESCRIBE	C→S
GET_PARAMETER	C→S, S→C
OPTIONS	C→S, S→C
PAUSE	C→S
PING	C→S, S→C
PLAY	C→S
REDIRECT	S→C
SETUP	C→S
SET_PARAMETER	C→S, S→C
TEARDOWN	C→S

IV. Tendencias Actuales

Actualmente el streaming de vídeo y audio es una de las aplicaciones más populares de la red. Hoy en día cualquier particular puede retransmitir audio y/o video, almacenado o en vivo, a través de la red. Esta posibilidad también es aprovechada, por supuesto, por emisoras de radio y televisión "tradicionales". No obstante, y a pesar del potencial del streaming en tiempo real, una de las mayores revoluciones experimentadas en internet tiene que ver con la creación y almacenamiento on-line de estos contenidos multimedia.

La llamada Web 2.0 permite a los usuarios compartir sus propios contenidos con los demás internautas. Populares portales como YouTube o Metacafe son, con toda seguridad, de las webs que más tráfico atraen diariamente en todo el mundo. Este tipo de portales y otras webs que puedan hacer del streaming de contenidos multimedia almacenados su reclamo, generalmente ya no utilizan la tecnología RTSP que ha quedado relegada casi exclusivamente para aplicaciones de retransmisión en riguroso tiempo real. Actualmente la tendencia es utilizar la tecnología Flash de Adobe.

A. Streaming Flash Movies

La mencionada tecnología Adobe Flash nos proporciona dos modos de hacer streaming:

- "Streaming verdadero", que no es más que un protocolo similar al RTSP pero propietario. Esto unido al hecho de que no ofrece grandes ventajas sobre el "tradicional" RTSP y que solo se puede emplear con la herramienta de pago Flash Communication Server (FCS) hace que esta modalidad no sea muy popular.
- *"Progressive Download"* o Descarga Progresiva. Es la técnica más utilizada de las dos, ya que no necesita ningún tipo de soporte especial del servidor y su uso es gratuito.

1) ¿Cómo funciona?

Para la difusión mediante streaming de contenido multimedia por el método de descarga progresiva se necesitan dos ficheros:

- Una pequeña película Flash (con extensión .swf) que simplemente será el marco donde se visualizará la información transferida y los controles para interactuar con la reproducción.
- El vídeo y/o el audio a transmitir codificado en un fichero con extensión .flv. Este archivo admite diferentes códecs de compresión de audio/vídeo, permitiendo un amplio abanico de calidades para adecuarse a las diferentes velocidades de conexión.

Como se ha mencionado anteriormente el método *"Progressive Download"* es el método más utilizado de los dos que nos ofrece la tecnología Adobe Flash, aunque no es realmente streaming.

La descarga progresiva funciona creando un buffer en el cliente y a medida que se va haciendo el volcado de datos al mismo, se va reproduciendo el contenido ya almacenado.

2) Pros

- Para utilizar esta tecnología no es necesario ningún soporte especial del servidor. El único requisito es que el cliente tenga instalado el plug-in adecuado en su navegador web.
- Las "descargas progresivas" se transmiten utilizando el protocolo HTTP que garantiza el correcto envío de todos los paquetes y, por tanto, asegura la retransmisión de la película entera sean cuales sean las condiciones de la red.
- El hecho de ser tráfico web estándar nos evita que el firewall (si lo tuviera) del cliente nos pueda bloquear el streaming.
- Además ya que, a mayor o menor velocidad, se tiene que terminar por descargar todo el contenido, podemos estar seguros de que todos los clientes disfrutaran de la misma calidad de reproducción, independientemente de la velocidad de su conexión a internet.
- Esta última característica nos permite utilizar algoritmos de compresión con mayor calidad que los que usaríamos en el caso de streaming "tradicional".

3) Contras

Las principales pegas que se le pueden achacar a este tipo de difusión vienen de la necesidad de descargar el contenido entero a la memoria caché del cliente.

- La cantidad de memoria que se tiene que reservar en el cliente para el buffer tiene que ser igual a la del contenido a reproducir.
- Para asegurar una reproducción sin cortes se necesita un tiempo de predescarga que será mayor cuanto mayor sea el contenido a reproducir.
- No se puede adelantar la película hasta el punto que nos interese mientras no se haya descargado al buffer, todo el contenido anterior a ese punto.
- Todos estos motivos hacen de ésta una solución poco práctica para la reproducción de contenidos de larga duración, siendo solamente atractiva para clips de audio o de video de corta duración, como trailers de películas, spots publicitarios, etc.

B. Streaming FLV mediante PHP

La solución a buena parte de estas trabas llegó en 2005 por parte de un grupo de usuarios insatisfechos con todas las soluciones de streaming existentes hasta la fecha y que querían alcanzar una solución satisfactoria antes de que las grandes multinacionales del sector "impusieran" una solución de código propietario [6].

El proceso se basa en una idea muy sencilla. Consiste en una descarga progresiva de una película Flash con extensión .flv, igual que hemos explicado antes, pero fragmentando la información con marcas temporales e introduciendo fotogramas clave en la película. Un script PHP se encargará de la comunicación entre cliente y servidor.

Dicho script además se encargará de gestionar la tasa de transferencia en tiempo real. Por ejemplo, aunque el buffer en principio bastaría que fuese tan grande como el mayor fragmento que haya entre fotogramas clave, si nos "sobrara" ancho de banda y memoria caché, se podría crear un segundo buffer donde ir almacenando los siguientes fragmentos de película por si para entonces las condiciones de la red no nos son tan favorables.

Esta solución también nos permite adelantar contenidos. Al hacer click en un punto posterior de la película se ejecutará una función que demandará al servidor la marca temporal que está reclamando el cliente en el reproductor de su navegador. Desde ese momento el servidor enviará al buffer del cliente información solo a partir del fotograma clave más cercano al solicitado, evitando así que se tenga que descargar toda la película hasta ese punto.

Esta solución, aunque sigue sin ser auténtico streaming, sí que consigue acercarse mucho, conservando además todas las ventajas de la descarga progresiva. Así pues, no es de extrañar que se haya popularizado tanto el streaming de ficheros flash mediante script PHP, hasta el punto de que el RTSP hoy en día ha quedado relegado casi exclusivamente a aplicaciones auténticamente en tiempo real, como televisiones o radios por internet.

Actualmente esta tecnología está registrada bajo una licencia Creative Commons (Reconocimiento 2.0 Inglaterra y País de Gales) [7] por Stefan Richter.

C. Comparativa

A modo de resumen adjuntamos dos tablas [8] en las que se comparan las características y aplicaciones de cada uno de los métodos de streaming vistos en este apartado IV.

TABLA III

APLICACIONES ADECUADAS PARA CADA TIPO DE STREAMING

Tipo de Aplicación	**Streaming**	**Descarga Progresiva**	**PHP Streaming**
Retransmisión en directo	X		
WebTV	X		X
E-Learning	X		X
Presentación on-line	X		X
Trailers de películas		X	X
Otros clips de corta duración		X	X

TABLA II

CARACTERÍSTICAS DE LOS DISTINTITOS MÉTODOS MENCIONADOS

Características	**Streaming**	**Descarga Progresiva**	**PHP Streaming**
Retransmisión en tiempo real	X		
Clips de larga duración	X		X
Acceso inmediato a diferentes puntos de una película			X
Descarga la película entera		X	
Descarga la parte de la película que sea necesaria			X
El FLV se guarda en caché del cliente		X	
Necesita un servidor especializado para streaming	X		
Necesita un servidor web con PHP			X
Puede ser parado por Firewalls	X		
Reproducción de alta calidad con cualquier conexión		X	X
Retransmite paquetes perdidos		X	X

REFERENCIAS

[1] Entrada en la wikipedia castellana sobre la Ley de Moore: http://es.wikipedia.org/wiki/Ley_de_Moore

[2] Historia del formato mp3 disponible en: http://www.pcdoctor.com.mx/Radio%20Formula/temas/Historia%20del%20MP3.htm

[3] Real Time Streaming Protocol Internet-Draft. Internet Engineering Task Force (IETF). Disponible en: http://ietfreport.isoc.org/allids/draft-ietf-mmusic-rfc2326bis-03.pdf

[4] Entrada en la wikipedia inglesa sobre el multicasting: http://en.wikipedia.org/wiki/Multicast

[5] Definición de unicast en la wikipedia inglesa: http://en.wikipedia.org/wiki/Unicast

[6] Sitio web donde se originó el streaming de ficheros .flv con código PHP: http://www.flashcomguru.com/index.cfm/2005/11/2/Streamingflv-video-via-PHP-take-two

[7] Licencia Creative Commons 2.0 Reconocimiento 2.0 Inglaterra y País de Gales, en castellano: http://creativecommons.org/licenses/by/2.0/uk/deed.es

[8] Tablas extraídas y traducidas de la web: http://www.rich-mediaproject.com/

Asignatura impartida por: Magdalena Payeras Capellà

Nombre: Samuel Reaño Montoro.
samuel.r.montoro@gmail.com
Titulación: Ingeniería Técnica de Telecomunicaciones, especialidad Telemática.

Nombre: David Mateos Costilla.
david.ptm56@gmail.com
Titulación: Ingeniería Técnica de Telecomunicaciones, especialidad Telemática.

Asignatura impartida por: Magdalena Payeras Capellà

Rehabilitació amb màxima eficiència energètica i arquitectònica

Margalida Mas Mesquida i Caterina Pinya Oliver.

Arquitectes Tècnics
masmespinya@hotmail.com

El nostre Treball Final de Carrera el varem presentar el novembre de 2006 i va ser un dels deu primers que es defensaren en aquesta universitat, ja que formarem part de la primera promoció d'Arquitectura Tècnica de la UIB que començà el curs 2002-2003. Duia per títol "Rehabilitació amb màxima eficiència energètica i arquitectònica" i va ser guardonat amb el 1er premi de la I Convocatòria de Premis a Projectes de Fi de Carrera dels estudis d'Arquitectura Tècnica de la UIB, que atorga la Càtedra Fundació Miquel Llabrés Feliu.

XIX. Objectiu Del Projecte

L'Objectiu del nostre projecte era, a partir de la reforma d'una edificació situada a Campos, determinar possibles actuacions bioclimàtiques (actives i passives) per a realitzar una construcció més sostenible, com a resultat de l'aplicació de criteris d'estalvi energètic. Varem tenir en compte també la supressió de barreres arquitectòniques en l'edifici.

En la rehabilitació convertirem l'edifici en un Casal de Joves. D'aquesta manera, cobríem una sèrie de demandes del poble, a més de treballar per a la integració dels joves amb qualsevol tipus de discapacitat i conscienciar de la problemàtica mediambiental afavorint la implementació de mesures d'estalvi energètic.

Fig. 2 Vista de la biblioteca del Casal de Joves un cop rehabilitat. 3D realitzat amb el programa Sketch-up.

XX. L'Entorn En La Rehabilitació

És molt important analitzar l'entorn de les edificacions per a dur a terme una rehabilitació eficient, ja que aquestes, moltes vegades, no responen ambientalment a les característiques del lloc on estan ubicades. Com bé explica l'arquitecte Glenn Murcutt: *"Segons la meva opinió, és una bogeria construir on no coneixes ni la llengua, ni cultura, ni la psicologia. Entendre la meva feina és fer arquitectura amb responsabilitat i per això s'ha de conèixer on construeixes. Has de conèixer la seva geologia, hidrologia, flora i clima. Quan Utzon va projectar l'Òpera de Sydney es va traslladar a viure allà."*

E. Confort

Entre els objectius d'aconseguir un recondicionament sostenible, destaca la salut i confort dels ocupants. Per confort entenem l'estat ideal d'una persona en una situació de benestar i comoditat. Els paràmetres que hi influeixen són la temperatura de l'aire, la humitat relativa, la radiació solar... Són factors que depenen directament del clima.

F. Clima

El clima caracteritza una regió, influeix en l'edifici i el seu comportament i sobretot en l'home i el seu confort. Per fer l'estudi de l'entorn de l'edifici varem utilitzar un programa anomenat Archisun, que és una eina que avalua el comportament tèrmic de l'edifici, tenint en compte la ubicació, entorn, forma, pell i interior de l'edifici. Obtenint així unes gràfiques de comportament estacional de l'edifici.

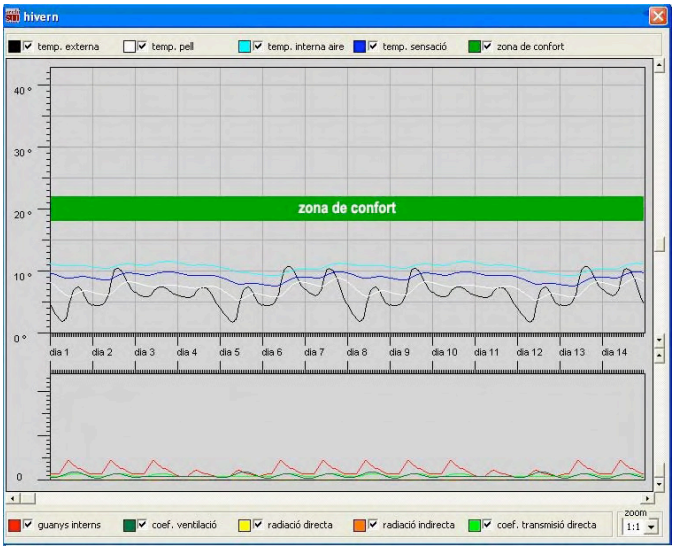


Fig. 2 Les gràfiques ens indiquen la temperatura de sensació dins l'edificació (entre d'altres) i ens permeten comparar amb la zona de confort tèrmic.

La conclusió que extraiem d'aquest anàlisi és que hem d'aïllar convenientment l'edificació i implementar instal·lacions eficients. Tot això concorda amb l'objectiu final del nou CTE.

XXI. Eficiència Energètica

Cercàvem una solució alternativa que ens donés màxima eficiència energètica, és a dir, la reducció del consum d'energia mantenint els mateixos serveis energètics, sense disminuir el confort i la qualitat de vida, protegint el medi ambient, assegurant l'abastiment i fomentat el comportament sostenible en l'ús. *Hem de tenir en compte que els edificis*

consumeixen el 50% de l'energia utilitzada per l'home, i produeixen la meitat de les emissions de CO2 a l'atmosfera.

XXII. ARQUITECTURA BIOCLIMÀTICA

De l'estudi del confort tèrmic dins les edificacions, en deriva tot un pensament arquitectònic que es denomina Arquitectura Bioclimàtica.

L'arquitectura Bioclimàtica és aquella que el seu disseny s'optimitza per aprofitar el clima i les condicions de l'entorn, amb la finalitat d'aconseguir una situació de confort tèrmic en el seu interior.

Un bon disseny bioclimàtic pot aconseguir un estalvi de fins el 70% per a la climatització i il·luminació de l'edifici. Tot això amb un increment del cost de construcció no superior al 15% sobre el cost estàndard.

Per aplicar la solució alternativa, es tenen en compte sistemes passius i actius.

G. Sistemes Passius

Els sistemes passius consisteixen en adequar el disseny arquitectònic de l'edifici al seu balanç tèrmic. S'utilitzen per fer-ho més eficient enfront a les inclemències climàtiques. Alguns exemples d'aquest sistemes són:

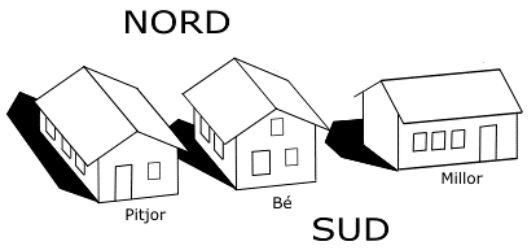


Fig. 3 Orientació de l'edifici.

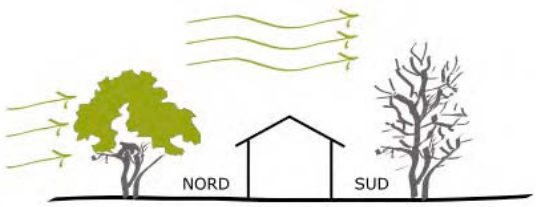


Fig. 4 Ventilació de l'edifici.

El sistema passiu per excel·lència és l'aïllament, quan aquest és adequat estalviam energia i doblers, però no s'han d'assolir gruixos d'aïllament exagerats, que no es puguin amortitzar energèticament amb un termini normal.

Des del punt de vista de la sostenibilitat, és fonamental conèixer la qualitat biològica dels materials, composició natural amb absència de substàncies nocives, valors que són difícils de trobar en els aïllaments sintètics però que sí es troben en els *biomaterials* (són naturals, no tòxics, produeixen baix impacte ambiental, són d'origen local, transpirables...) també destaca la seva vida útil, ja que es poden reciclar o reutilitzar.

Per aïllar les parets de l'edifici varem usar cel·lulosa insuflada (provinent del reciclatge de paper de diari) dins un trasdosat. Per a aïllar coberta i forjats usàrem panells de fibra de fusta i per impermeabilitzar, una làmina de cel·lulosa.

Per a la col·locació dels sistemes passius fou necessari la realització d'un estudi d'assolellament, és a dir, analitzarem diferents hores de distints mesos per tal d'esbrinar les zones amb màxima incidència de llum solar.

Fig. 5 Vista de les cobertes del Casal amb la incidència de llum solar.

No existeix una única solució a l'hora d'aplicar Sistemes Passius, la idoneïtat de les propostes pot variar segons l'època de l'any, a més, ens limitava el fet de no poder actuar en l'orientació de l'edifici.

La vegetació és un excel·lent dispositiu de control tèrmic, per evitar el malbaratament d'aigua varem utilitzar la xerojardineria, que es basa en l'ús de les espècies autòctones o adaptades que es caracteritzen per la seva resistència al clima cíclic i extrem.

H. Sistemes Actius

Els sistemes actius són tecnologies que es poden introduir a l'edifici per reforçar els beneficis aconseguits amb els sistemes passius. L'energia és un recurs clau en el procés de la construcció sostenible.

Existeixen les energies No Renovables, bàsicament combustibles fòssils, que tenen l'avantatge de no necessitar energia de suport pel seu funcionament, però són limitades, a més de generar emissions i residus.

Les energies Renovables com la solar, eòlica o la geotèrmica, són netes, il·limitades, però actualment necessiten suport de les energies No Renovables.

Les energies Renovables tan sols representen un 9.6% mentre que les energies No Renovables més usades a Espanya són el petroli i el gas, que configuren un 75.9% del total.

L'esquema definitiu a seguir per l'aplicació de sistemes actius renovables al Casal fou: aigua calenta amb bomba de calor, sòl radiant i energia solar tèrmica.

- Bomba de calor, és una màquina tèrmica que permet transferir calor d'una font freda a una altra més calenta. És un aparell molt eficient ja que és capaç de subministrar més energia tèrmica de l'elèctrica que consumeix. És reversible, és a dir, es pot usar tot l'any.
- Sòl radiant, està format per un conjunt de canonades que passen per davall del sòl de l'edifici, a través del qual circula un fluid que pot ser calent o fred. Calefactant sobre el sòl i no al sòtil, amb el conseqüent estalvi energètic, s'apropa a la climatització ideal.

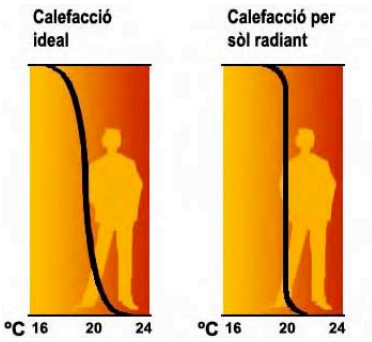


Fig. 6 Comparativa de corbes de calefacció.

- Energia solar per ACS i pel recolzament de la calefacció amb bomba de calor.

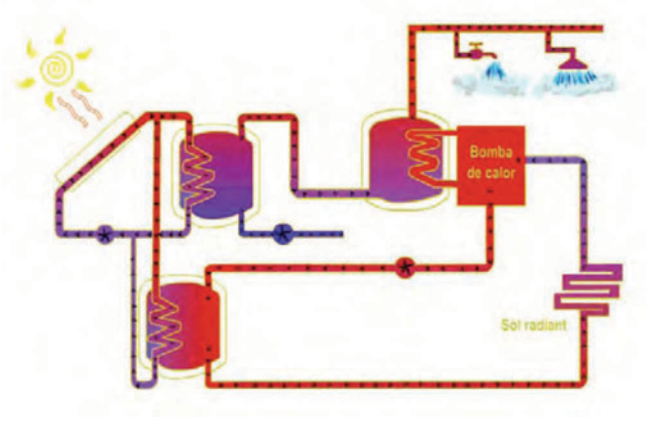


Fig. 7 Esquema de la instal·lació eficient del Casal.

XXIII. Comparativa De Solucions

Una part important del projecte consistia en fer la comparativa de l`estalvi energètic de la solució alternativa, anteriorment exposada, amb una solució convencional, és a dir, la que es duu a terme en qualsevol obra avui en dia.

Varem fer els càlculs dels consums anuals de les dues instal·lacions de climatització a partir de les calories i frigories necessàries a cada estància per a aconseguir la temperatura de confort.

El preu de la solució alternativa tant en temes passius com d'instal·lacions, és més car. Però per jutjar la seva viabilitat ens cal saber el seu període d'amortització.

La inversió inicial de la solució alternativa és substancialment major, però ja que els consums en climatització són bastant més reduïts, aquesta solució s'amortitza en uns 9 anys.

Consideram que és un bon termini ja que el retorn de la inversió ha de ser proporcional a la vida útil de la instal·lació.

XXIV. Conclusió

La conclusió més important d'aquest projecte és que varem comprovar que és possible i viable rehabilitar amb sistemes alternatius avui en dia, a Mallorca, i recuperar la inversió en períodes inferiors a 10 anys.

Per últim dir que al començament del projecte érem bastant escèptiques enfront als biomaterials i a les energies alternatives, però aquest ens ha permès aprofundir en el tema i sobretot comprovar l'efectivitat tant dels materials com de les instal·lacions.

Agraïments

A Joan Muñoz, director del projecte. A Lleonard Borràs, de Sol i Clima i a Poncio Ripoll de Bioconstrucció. A tots aquells que ens varen recolçar en el seu moment. I a en Toni Cladera per donar-nos la oportunitat de reviure el projecte (per tercera vegada).

Referències

[8] Fullana, M. (s/d), *Diccionari de l'art dels oficis de la construcció*, editorial Moll.
[9] Garcia-Delgado, C. (s/d), *La casa popular Mallorquina*, editorial La Foradada.
[10] Rodríguez Viqueira (s/d), *Introducción a la Arquitectura Bioclimàtica*, Noriega editores.
[11] D.K. Ching, F. (s/d), *Diccionario visual de arquitectura*, edición GG.
[12] http://www.ecohabitar.org
[13] http://www.soliclima.com
[14] http://www.construible.es
[15] http://www.bioma.com
[16] http://www.bioconstruccion.biz

Diseño de un Registro de Desplazamiento en Tecnología CMOS

Fanny Serapio Fernández , Bartomeu Obrador Barceló

fanny.sf@gmail.com, tomeu1985@gmail.com

"Sistemes Microelectrònics", asignatura optativa de Enginyeria Tècnica en Telecomunicacions, Enginyeria Tècnica Industrial i Física.

Este artículo describe las principales etapas para la implementación del componente SN74164 (Registro de desplazamiento de 8 bits) utilizando una tecnología CMOS de 0.35µm.

1. Introducción

El componente SN74164 es un registro de desplazamiento de 8 bits basado en *flip-flops* tipo D. Un *flip-flop* D es un circuito de memoria biestable que utiliza una señal de sincronismo para especificar cuando la memoria mostrará a la salida la señal de entrada (transparencia) y cuando la memoria no responderá a los diferentes cambios de la señal entrante (no transparencia) [1].

A su vez, un *flip-flop* D está formado por diferentes puertas lógicas: puertas "tristate" que, gracias al valor de alta impedancia (Z), son útiles para transferir datos de una parte del sistema al otro; puertas de "paso" que nos permiten transmitir satisfactoriamente ambos valores lógicos (0 y 1).

Para poder implementar estos circuitos digitales de forma física, se usan transistores MOS. La tecnología CMOS permite implementar dos tipos de transistores (nMOS y pMOS) en un mismo circuito integrado. Los transistores MOS utilizan un único tipo de carga eléctrica para transferir información: los transistores n utilizan electrones (polarización negativa) mientras que los transistores p utilizan "huecos" (un "hueco" equivale a la ausencia de un electrón y tiene polaridad positiva) [3]. Dicha carga circula entre los terminales de fuente y drenador de cada transistor, en función del estado lógico de su terminal de puerta.

El objetivo de dicho trabajo ha consistido en implementar un registro de desplazamiento a partir de *flip-flops* D y ver su funcionamiento, así como los diferentes retardos de puerta que presenta. De esta forma nos permitirá analizar el comportamiento de un *flip-flop* tanto de forma singular como en agrupación, pudiendo así determinar sus posibles ventajas o inconvenientes.

Figura 1. Transistor MOS discreto

2. Metodología y diseño

Un registro de desplazamiento es un elemento secuencial formado por 8 *flip-flops* D. Cada *flip-flop* D tiene conectado en paralelo una señal de Reset asíncrona (para "inicializar" los registros) y una señal de reloj que hará que el *flip-flop* se active por flanco de subida. Tenemos dos señales de entrada (A y B) que, al aplicarles una puerta NAND configuran una única señal de entrada al circuito (Figura 2).

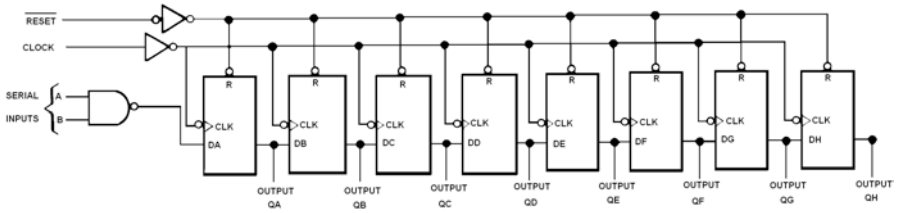


Fig. 2 Diseño digital del registro de desplazamiento

Para realizar el circuito digital, hemos realizado un diseño microelectrónico utilizando el programa *Microwind*. Este programa permite representar circuitos de forma gráfica en el nivel físico (dichas representaciones se denominan *layouts*). Mediante este programa se han realizado los diferentes *layouts* de cada puerta distinta mostradas en la figura anterior. En el caso del *flip-flop* D, está compuesto por una puerta NAND de 2 entradas, una puerta NAND tristate, un inversor y una puerta de paso [2].

Una vez diseñada cada puerta de forma individual, hemos empezado a conectar todos los elementos entre sí siguiendo el esquema eléctrico de la figura 3.

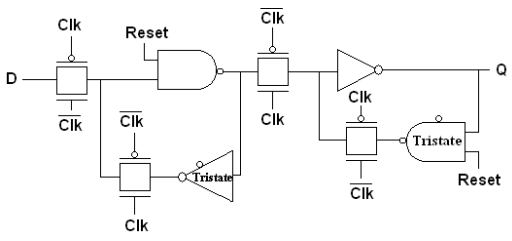


Figura 3 Esquema eléctrico de un *flip-flop*

Para el diseño de cada puerta hay que tener en cuenta que cada una de ellas tiene, como mínimo, un transistor nMOS con su difusión n y otro transistor pMOS con su pozo y su difusión p. El polisilicio es el material encargado de definir la ubicación de los transistores y conducir las diferentes señales que conectemos a los transistores p o n.

Para conectar los diferentes transistores es necesario el uso de varios niveles de metal (niveles 1, 2 y 3). El primer nivel de metal puede unirse con el polisilico o con los terminales de fuente y drenador de los transistores mediante el uso de contactos, el resto de niveles de metal pueden unirse entre sí mediante el uso de vías.

Una vez conectadas las puertas entre sí, éstas deben compartir el mismo pozo, la misma fuente de alimentación y la misma toma tierra, para obtener más sencillez y, a su vez, eficiencia.

La tecnología que hemos utilizado es una tecnología CMOS de 0,35µm con tres niveles de metal, esto implica que la longitud de canal de los transistores no puede ser inferior a 0,35 µm. Una vez conectado todo el sistema que compone un *flip-flop* (Fig. 4), hay que conectar las señales comunes que tienen todas las puertas (la señal de reloj "Clk" y la de "Reset").

El siguiente paso a realizar es simular el conjunto que hemos creado para asegurarnos que su funcionamiento es el correcto. En dicha simulación, las señales que se asignan a las entradas del *flip-flop* pueden ser pulsos, señales de reloj y/o valores lógicos constantes (voltaje alto "1" o bajo "0"). Las salidas únicamente serán nodos donde visualizaremos los resultados. El resultado de la simulación consiste en una gráfica donde se pueden observar los diferentes cambios que ha sufrido el sistema, tanto en las entradas como en las salidas. Para nuestro caso, mientras que la señal de entrada "D" tenga el valor 0, el *flip-flop* se comportará de manera no-transparente. Cuando la señal D tenga el valor 1, la salida tomará el mismo valor que la entrada D una vez que haya un flanco de subida (por la señal de reloj Clk).

Finalmente, calcularemos el tiempo de *set-up*, definido como el mínimo intervalo de tiempo entre una transición en la entrada y el flanco de subida del reloj de manera que el *flip-flop* consiga capturar el nuevo valor asignado a la entrada, y el tiempo *hold* o intervalo mínimo de tiempo en que la entrada debe mantener su valor después de una transición de flanco bajada de reloj para que la salida del *flip-flop* no modifique su valor [3].

3. Implementación y resultados

Una vez que hemos comprobado que el *flip-flop* funciona correctamente, llega el momento de realizar el *layout* del registro completo, para ello hay que conectar los 8 *flip-flops* en serie, tal como se ha indicado en la figura 2, junto con las señales Clk y reset paralelamente (Fig. 5).

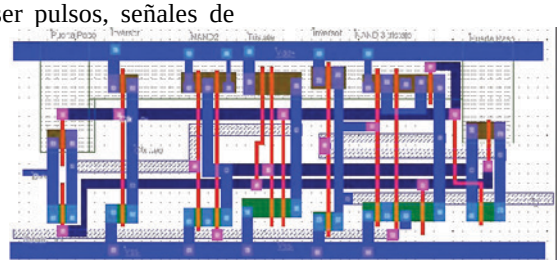


Fig. 4 *Layout* de un *flip-flop* realizado con *Microwind*

Se van uniendo los diferentes *flip-flops* hasta obtener un circuito con 8 *flip-flops*. La salida de cada uno de ellos se conecta con la entrada del siguiente, para obtener así la cadena de desplazamientos.

Una vez simulado, obtenemos la gráfica de la figura 6. En ella se puede observar como los diferentes *flip-flops* van desplazando la señal D. Únicamente el *flip-flop* D trabajará en estado transparente si las señales A y B tienen valor 1.

Cabe destacar que al inicio de la simulación se produce una irregularidad, eso es debido a que el sistema necesita un tiempo de estabilización. También se puede observar en la gráfica 6 que la respuesta (Q) de cada *flip-flop* presenta un cierto declive (*glitch*), eso es debido a las puertas de paso captan las transiciones más lentamente que las otras puertas y por eso se produce dicho fenómeno.

Finalmente, para determinar el tiempo de *set-up* hemos ido variando el instante en que la señal de entrada cambia de valor, aproximándolo cada vez más al valor del flanco de subida del reloj Clk, comprobando que el sistema siga funcionando correctamente. En nuestro caso, hemos determinado que el tiempo de *set-up* es de 100 ps, este valor es función de la tecnología elegida, y del ancho y longitud de canal de los transistores utilizados.

Para determinar el tiempo de *hold*, hemos seguido el mismo procedimiento pero esta vez hemos variado el valor del pulso de la entrada D, una vez ya producida la transición en el reloj, hasta llegar al valor más próximo del flanco de bajada del Clk y que, a su vez, el sistema también siguiera funcionando correctamente. El valor obtenido es de 94 ps.

Una vez que se superaban estos valores el sistema dejaba de funcionar correctamente, es decir, no obteníamos la respuesta esperada en la salida del *flip-flop*.

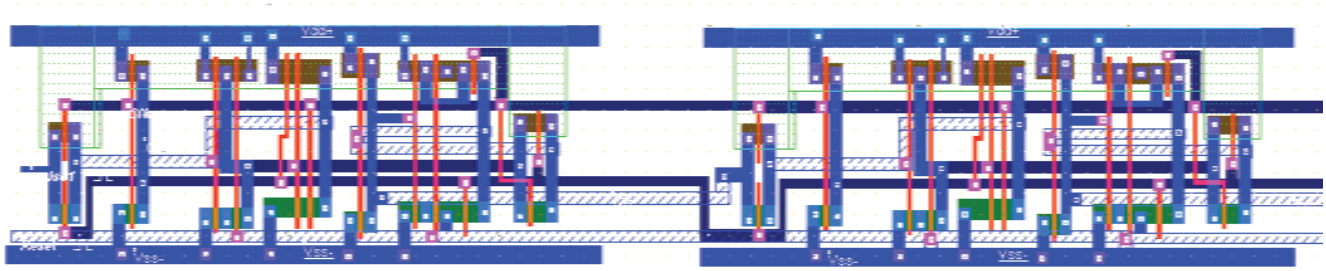


Fig. 5 Conexión de dos flips flops

Fig. 6 Resultados obtenidos tras la simulación eléctrica del registro de 8 bits.

4. Conclusiones

Después de los resultados obtenidos podemos comprobar como se puede construir un elemento complejo (registro de desplazamiento) a partir de una unidad más simple, el *flip-flop* D. También ha sido útil este trabajo para la investigación del comportamiento de las puertas tristate y las puertas de paso.

Cabe destacar que el programa *Microwind* nos ha sido de gran utilidad ya que nos ha permitido hacer una simulación "virtual" del circuito. Esta simulación nos ha permitido analizar el comportamiento del mismo y, a su vez, obtener y mejorar los resultados esperados sin tener que recurrir a la implementación física de los transistores CMOS.

En cuanto a las aplicaciones al mundo real, pueden ser muy diversas: un decodificador para uso en domótica, un contador digital, contador en anillo, etc.

Bibliografía

[1] John P. Hayes , *Introduction to Digital Logic Design*, 1st ed., Addison Wesley, 1993
[2] A. Rubio, P. Altet, X. Aragonés, J González, D. Mateo y F. Moll. *Diseño de Circuitos y Sistemas Integrados*. Edicions U.P.C.
[3] Barry Wilkinson, *Digital System Design*, Longman Higher Education, September 1986.
[4] Manual *Microwind Lite*

Biografía

- Fanny Serapio nació en Capdepera el año 1986. Es alumna de tercer curso de Ingeniería Técnica en Telecomunicaciones (especialidad en Telemática) del Colegio Politécnico Superior de la UIB.

- Bartomeu Obrador nació en Cala Figuera el año 1985. Es alumno de tercer curso de Ingeniería Técnica en Telecomunicaciones (especialidad en Telemática) del Colegio Politécnico Superior de la UIB.

Diseño e implementación de un núcleo para aplicaciones de tiempo real

Alberto Ballesteros Varela, Bartolomé Palmer Riera

Disseny de Sistemes Operatius, 5º Enginyeria Informàtica

Sumario— **Este documento describe el diseño y la implementación de un núcleo de sistema operativo que da soporte a aplicaciones de tiempo real. El núcleo consiste en un planificador de tareas híbrido que proporciona mecanismos para la ejecución de tareas concurrentes en tiempo real. El núcleo pone a disposición de las tareas primitivas de exclusión mutua; primitivas de sincronización periódicas y aperiódicas; así como una interfaz de entrada y salida de datos adecuada.**

I. INTRODUCCIÓN

Un sistema operativo (SO) es un conjunto de programas que se encarga de gestionar los recursos de un ordenador y proporcionar una interfaz con el usuario. Algunos de los sistemas operativos modernos más conocidos son Windows, MacOSX o Linux.

Algunos sistemas operativos dan soporte a sistemas concurrentes, en el sentido que permiten ejecutar varias tareas (programas) de forma virtualmente simultánea. Esto se consigue asignando continuamente la unidad central de proceso (CPU) a cada tarea durante un corto periodo de tiempo.

La concurrencia aporta muchas ventajas, pero también plantea ciertos problemas. Uno de ellos es la sincronización entre tareas. Es necesario proporcionar herramientas que permitan a las tareas sincronizarse y coordinarse en la consecución de un objetivo común.

Otro problema que debe abordarse en sistemas concurrentes se deriva del uso de recursos compartidos. Para cada recurso es necesario garantizar que únicamente un número limitado de tareas (normalmente sólo una) acceda a él simultáneamente. De esta forma, se define el concepto de *región crítica* (RC) como: parte de una tarea, asociada a un recurso, que debe ser realizada sin que otra tarea pueda acceder al mismo. Se dice también que dentro de una región crítica hay *exclusión mutua*, puesto que el acceso de una tarea excluye al resto.

Una de las aplicaciones más importantes de los sistemas concurrentes es el control de *sistemas de tiempo real*. Un *sistema de tiempo real* (STR) es aquel que debe producir resultados correctos cumpliendo plazos de tiempo específicos. Estos sistemas están presentes en escenarios muy diversos: comunicaciones, control de vuelo y satélites, robótica, centrales nucleares, etc. En muchas de estas aplicaciones, un fallo de funcionamiento debido a la violación de un plazo temporal puede tener consecuencias catastróficas.

Un sistema operativo de tiempo real (SOTR) es pues un SO que proporciona mecanismos para que las tareas puedan ejecutarse concurrentemente respetando sus plazos temporales. El respeto de estos plazos es de vital importancia, puesto que en algunos STR basta que una tarea no termine su trabajo dentro de su plazo, para que se considere que el sistema ha fallado.

La entidad más básica de un sistema operativo es el núcleo o *kernel*. El núcleo gestiona la ejecución de las tareas, el uso de la memoria, y provee funciones de entrada/salida de datos a bajo nivel.

Existen diferentes arquitecturas de SO. En esta práctica hemos adoptado una arquitectura de tipo *microkernel*, que consta de un núcleo pequeño optimizado, dónde los añadidos se suman como módulos independientes. En nuestro caso sólo nos interesan las restricciones temporales de las tareas. Por tanto, nos hemos centrado en la implementación de la parte del núcleo que gestiona la ejecución de las tareas: el *planificador*. Adicionalmente, hemos implementado algunas funciones de entrada y salida de datos.

Nuestro SOTR es capaz de dar soporte a la ejecución de un conjunto de tareas concurrentes mediante sencillas llamadas al núcleo. Para demostrarlo, hemos implementado un juego construido a partir de tareas concurrentes que se ejecutan sobre nuestro SOTR.

II. OBJETIVOS

Nuestro objetivo es diseñar e implementar un núcleo de SO para aplicaciones de tiempo real. La parte principal es el planificador, que debe encargarse de decidir cuál es la tarea más prioritaria que debe ser ejecutada y de asignar la CPU a esta nueva tarea.

El planificador debe ser híbrido. Es decir, el planificador debe ejecutarse periódicamente y, además, debe ejecutarse para responder a eventos externos aperiódicos, p.e. el pulsado de una tecla.

Un evento externo genera lo que se denomina una *interrupción*. Ésta es una condición que hace que la CPU interrumpa momentáneamente el trabajo que estaba haciendo y ejecute una función conocida como *rutina de servicio a la interrupción* (RSI). El SOTR debe proporcionar al programador una primitiva, que llamaremos *hook*, para que éste pueda insertar una RSI que atienda al pulsado de cualquier tecla.

Además, el núcleo debe proporcionar a las tareas las siguientes primitivas de sincronización y exclusión mutua:

(1) Sincronización periódica mediante la primitiva *delayUntilTime(T)*. Suspende la ejecución de la tarea que la invoca durante *T* unidades de tiempo.

(2) Sincronización aperiódica mediante las primitivas de gestión de flags *waitFlag(flag, mask, condition, option) setFlag(flag, mask)* y *clearFlag(flag, mask)*. Un *flag* es básicamente una variable que contiene una condición. Una tarea que ejecute *waitFlag* se bloqueará hasta que el valor del flag sea el indicado por los parámetros *mascara* y *condición*. Cuando el flag contenga el valor por el cual una

tarea espera, la tarea podrá volver a ejecutarse y cambiará el valor del flag según lo indicado en el parámetro *option*. El valor contenido en un flag puede ser modificado por una tarea o por una RSI mediante las primitivas *setFlag* y *clearFlag*.

(3) Exclusión mutua mediante las primitivas de *semáforos binarios simples wait(S)* y *signal(S)*. Un semáforo binario básicamente es una variable binaria (su valor es *verdadero* o *falso*) que tiene una cola asociada y que protege una región crítica (RC). Una tarea que ejecute *wait(S)* sólo será capaz de entrar en la RC si el semáforo *S* está libre (la variable que indica el valor de *S* está a *falso*). Cuando esto ocurre, *S* pasa a estar ocupado (*verdadero*) por la tarea que hizo el *wait*. Por el contrario, si una tarea ejecuta *wait(S)* y *S* está ocupado, esta tarea se bloquea y espera en la cola del semáforo hasta que éste esté libre. Finalmente, cuando una tarea sale de la RC protegida por *S*, ejecuta *signal(S)*; esto libera a *S* de forma que alguna de las tareas que estaba en la cola de *S* accede a la RC y pasa a estar lista para ejecutarse de nuevo.

(4) Exclusión mutua mediante las primitivas de *semáforos binarios con techo de prioridad inmediato (IPCP) waitIPCP(S)* y *signalIPCP(S)*. Un semáforo IPCP funciona igual que uno simple. La diferencia estriba en que el semáforo IPCP tiene una prioridad asociada. Cuando una tarea consigue ocupar *S*, la tarea adquiere la prioridad de *S* hasta que libera a *S*. Este tipo de semáforo se utiliza para evitar que se produzca un *interbloqueo*: situación en que todas las tareas están bloqueadas esperando en la cola de algún semáforo. Debido a limitaciones de espacio, no entraremos en detalles de cómo se asignan las prioridades a los semáforos IPCP para evitar interbloqueos.

A parte de estas primitivas, el SOTR debe proporcionar a las tareas una función de salida por pantalla que sea adecuada para ellas. No se pueden usar las funciones de salida que proporcionan los lenguajes de programación convencionales, porque no están preparadas para ser usadas en sistemas concurrentes.

Finalmente, se debe desarrollar un juego de tenis simple (al que llamamos *Pong*) que se ha de ejecutar sobre el SOTR. Este juego estará compuesto por varias tareas periódicas y/o aperiódicas que irán interactuando para realizar toda la lógica y presentación, p.e. para mover la bola y las raquetas.

El sistema debe implementarse sobre una plataforma x86/MS-DOS, y deben desarrollarse con el compilador Borland C++ mediante el lenguaje C+

III. DISEÑO

El diseño está orientado a optimizar la gestión de las tareas. Una tarea es un conjunto de instrucciones que se ejecutan en orden secuencial. Cada tarea se encuentra en un determinado *estado*. La Figura 1 representa las transiciones entre estos *estados*, los cuales son:

- *Activo*. La tarea está lista para ser ejecutada.
- *Ejecución*. La tarea se está ejecutando actualmente en la CPU.
- *Suspendida*. La tarea no está lista para ejecutarse durante una determinada cantidad de tiempo.
- *Bloqueada*. La tarea no está lista para ejecutarse porque está a la espera de algún evento aperiódico (flag) o porque está esperando en la cola de un semáforo.

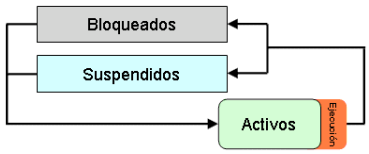


Fig. 1 Estados de una tarea y sus transiciones

Por otra parte, cada tarea tiene reservado un espacio de memoria, llamado pila, que tiene dos utilidades principales. (1) Permitir a la tarea guardar información sobre sus variables locales; (2) permitir a la tarea pasar parámetros y obtener resultados cuando realiza una llamada a función. Una función es un conjunto de instrucciones que realizan un cometido determinado que diferentes tareas podrían necesitar, p.e. varias tareas podrían invocar a una función llamada "calcularSen(2π) ", donde '2π' se pasaría como parámetro.

Además, la pila también permite guardar parte de la información que constituye el *estado de ejecución* de una tarea, cuando ésta es desplazada de la CPU. Es importante no confundir el *estado* de una tarea con su *estado de ejecución*. El *estado de ejecución* de una tarea consiste en el valor que tienen todos y cada uno de los registros de la CPU en un momento determinado de su ejecución, p.e. el valor del registro *program counter* indica cuál es la siguiente instrucción que la tarea va a ejecutar.

El hecho de que la pila permita guardar parte del *estado de ejecución* de su tarea es útil para el planificador. Cuando éste decide que una tarea diferente debe ocupar la CPU, realiza lo que se denomina *intercambio de contexto*. El *intercambio de contexto* consiste en guardar el *estado de ejecución* de la tarea que actualmente se está ejecutando, pero que va a ser desplazada, y restaurar el *estado de ejecución* de la nueva tarea que va a ocupar la CPU. Una vez hecho esto, el planificador indica a la CPU que tiene que seguir ejecutando la nueva tarea a partir del punto donde ésta se interrumpió cuando fue desplazada en el pasado. La Figura 2 muestra un ejemplo de este procedimiento.

Sin embargo, la pila no es suficiente para guardar toda la información necesaria para gestionar una tarea. Por ello, el planificador representa y guarda información adicional sobre una tarea determinada en una estructura de datos dedicada que se denomina *Task Control Block* (TCB).

Como veremos en la siguiente sección, los TCBs se agrupan en diferentes conjuntos, dependiendo del *estado* de las tareas que representan. Por ejemplo, habrá un conjunto que contendrá los TCBs de las tareas activas, otro con los TCBs de las tareas suspendidas, etc.

Un TCB concreto incluye la siguiente información:

- La pila de su tarea.
- Los datos sobre el *estado de ejecución* de su tarea que no pueden guardarse en la pila de ésta. Estos datos son imprescindibles para realizar el *intercambio de contexto*.
- La prioridad de su tarea.

- El instante de tiempo en el que su tarea debe despertarse, después de haber usado la primitiva *delayUntilTime.*
- La condición por la que su tarea está esperando para desbloquearse tras haber usado la primitiva waitFlag.

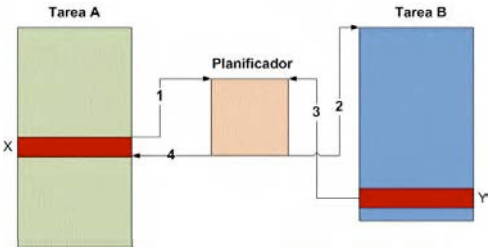


Fig. 2 Ejemplo de cómo dos tareas se alternan en la CPU

IV. IMPLEMENTACIÓN

Como se ha dicho antes, la información necesaria para gestionar una tarea se almacena en su TCB. Los TCBs se agrupan en diferentes conjuntos según el *estado* en el que se encuentren las tareas que representan. Cada conjunto de TCBs se implementa mediante la estructura de datos que más convenga. Pueden verse los conjuntos de TCBs en la Figura 3.

Básicamente hemos implementado dos tipos de estructuras: *lista simple* y *montículo*; si bien hemos utilizado también algunas variantes de la lista simple. Una lista simple es un conjunto de elementos no ordenados y almacenados de forma secuencial. Las inserciones se efectúan por el final y tienen un coste constante. Sin embargo, se necesita recorrer la lista para encontrar un elemento determinado almacenado en ella. Hemos implementado cada lista simple sobre un vector que tiene un tamaño máximo predefinido.

En un montículo los elementos se ordenan mediante una clave proporcionada en el momento de la inserción. Para mantener los elementos ordenados, éstos se organizan en forma de árbol binario. De esta manera, cada inserción tiene un coste logarítmico y las consultas tienen un coste constante. Un problema inherente a la implementación de esta estructura, es que sólo permite leer el elemento que tiene más prioridad.

La tabla I muestra los costes de las diferentes operaciones que se pueden realizar sobre estas estructuras.

TABLA II
COSTES OPERACIONES DE CADA ESTRUCTURA DE DATOS

	Inserción	**Eliminación**	**Consulta**
Lista simple	Constante	Constante	Lineal
Montículo	Logarítmico	Logarítmico	Constante

A continuación se describen los conjuntos de TCBs que se han implementado, así como la estructura de datos que se ha seleccionado para cada uno de ellos (Figura 3).

Es necesario remarcar que sólo *listaTarea* contiene TCBs; el resto de listas almacenan un apuntador (una referencia) a un TCB en concreto. De esta manera, sólo se almacena un TCB por tarea, con lo que se consigue ahorrar mucha memoria, sin pérdida de velocidad.

4) Conjunto de tareas (*listaTarea*): se implementa con una lista indexada por el identificador de la tarea. De esta manera, la lista permanece ordenada y el coste del acceso es lineal.
5) Conjunto de tareas activas (*activos*): está implementado con un montículo, ya que es necesario que estén ordenadas por prioridad.
6) Conjunto de tareas suspendidas (*suspendidos*): implementado con una lista simple, ya que es necesario recorrer todos los elementos.
7) Conjunto de tareas bloqueadas por diferentes semáforos (*listaBloqSem*): está implementado con una lista. Cada elemento contiene un montículo donde están almacenadas las tareas bloqueadas por un semáforo determinado.
8) Finalmente tenemos las tareas bloqueadas a causa de flags (*listaBloqFlag*): es el mismo caso que las tareas suspendidas. Se utiliza una lista simple, ya que es necesario recorrer todos los elementos de esta lista.

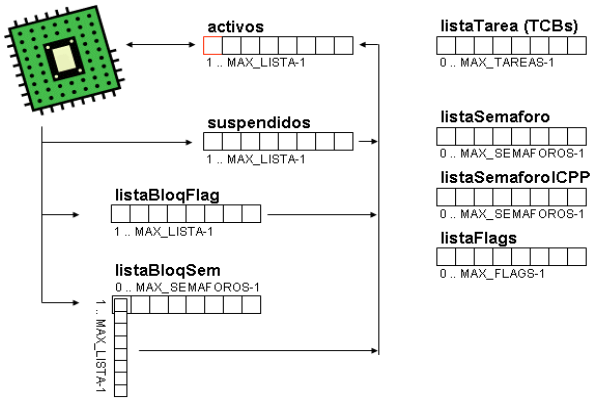


Fig. 3 Representación gráfica de los conjuntos de TCBs

Como se ha dicho anteriormente, el planificador utiliza estas estructuras y la información contenida en los TCBs para gestionar las tareas: para decidir qué tarea debe ejecutarse en cada momento y para realizar el *intercambio de contexto*.

La mayor parte del planificador está implementado dentro de una función, llamada *scheduler()*, que puede invocarse desde distintos puntos; básicamente, desde una tarea que ejecuta una primitiva de sincronización, o desde una rutina de servicio a la interrupción (RSI).

Sin embargo, parte del trabajo que realiza el planificador está implementado en las propias primitivas de sincronización. Es decir, una primitiva de sincronización puede realizar ciertas operaciones sobre los conjuntos de TCBs y, luego, invocar al *scheduler* para que realice el trabajo restante.

Notar que al estar ejecutándose en un entorno concurrente, una primitiva de sincronización tiene que asegurarse de que accede de forma exclusiva a los conjuntos de TCBs. Para ello utiliza una variable global que indica que está accediendo a datos del planificador. En la Figura 4 se ve un ejemplo de cómo se implementaría esta exclusión mutua. Cualquier RSI que quisiese invocar al *scheduler* debería verificar que la variable *planificadorOcupado* está a falso. Por tanto, al poner la variable *planificadorOcupado* a verdadero, la primitiva evita que una RSI pueda invocar al *scheduler* (el cual modificaría los TCBs) mientras ella los está manipulando.

```
primitiva_sincronización()
    {
       planificadorOcupado = verdadero
            acceso exclusivo a los conjuntos de TCBs
            si es necesario, llamar al scheduler
       planificadorOcupado = falso
    }
```

Fig. 4 Esquema de primitiva de sincronización para asegurar exclusión mutua

En cuanto a las rutinas de servicio a la interrupción (RSI), éstas sólo se ejecutan si ocurre cierto evento. Estos eventos están definidos físicamente en la arquitectura del propio ordenador. Cuando ocurre cierto evento se ejecuta su RSI asociada predeterminada.

Sin embargo, el ordenador permite redefinir la RSI que se ejecutará en cada evento. De esta manera, no sólo podemos instalar nuestras propias rutinas (RSIs del núcleo del SOTR), sino que podemos ofrecer al usuario instalar las suyas.

A las RSIs definidas por el usuario las denominamos *hooks*. Nuestro SOTR proporciona funciones para que el usuario pueda instalar y desinstalar *hooks* desde sus programas.

Por otro lado, un caso particular de RSI del núcleo del SOTR es la que controla el tiempo. Funciona usando un cronómetro de la propia arquitectura que tiene diferentes parámetros como el tipo de cuenta o el valor inicial. Variando estos parámetros se consigue que cada cierta cantidad de tiempo se ejecute nuestra RSI, que a su vez llama al *scheduler*. Esta RSI es muy importante; sin ella el SOTR no tendría un conocimiento exacto del tiempo que transcurre ni podría invocar al planificador periódicamente.

Finamente, se ha implementado el juego (*Pong*) que se ejecuta sobre el SOTR. Está compuesto por varias tareas, periódicas y aperiódicas, que irán interactuando para realizar toda la lógica y la presentación. Existe una tarea para cada una de las siguientes funcionalidades:

9) Gestionar el pulsado de cualquier tecla para actualizar las variables que determinan aspectos como: la posición de cada pala, la velocidad de la bola y la condición de finalización del juego.
10) Actualizar la posición de la bola según su dinámica.
11) Comprobar la posición de la bola y actualizar el marcador, si fuera necesario.
12) Refrescar el contenido que se visualiza en la pantalla.

Fig. 5 Instantánea de la pantalla del Pong

V. CONCLUSIONES

El núcleo es la parte más básica de un sistema operativo y, por tanto, la velocidad con que se ejecutan sus funciones es crucial a la hora de implementar servicios sobre él. Así pues, hemos diseñado e implementado el núcleo intentando optimizar su rendimiento. Por ejemplo, hemos puesto un cuidado especial en la elección de los tipos de datos y en la implementación del intercambio de contexto.

Además, nos hemos preocupado de generar un código modular. El sistema operativo está formado por entidades, lo más independientes posible, que cooperan. Esto hace que el sistema operativo sea más fácil de entender y mantener.

Esta práctica es una primera aproximación a lo que podría ser un sistema operativo comercial. Sin embargo, su realización nos ha ayudado a entender la problemática que se esconde bajo el diseño y la implementación de un planificador de tareas de tiempo real. Hemos desarrollado un tipo de sistema que hemos visto muchas veces funcionando, pero del cual no conocíamos los detalles internos. También nos ha dado la posibilidad de mejorar aspectos de una problemática base usando nuestros conocimientos e intuición.

Asignatura impartida por Manuel Alejandro Barranco González.

Alberto Ballesteros Varela es Ingeniero Técnico en Informática de Sistemas y actualmente está cursando Ingeniería Informática. Trabaja en el diseño, implantación y mantenimiento de sistemas de Terminales de Punto de Venta (TPV). *ballesteros.alberto@gmail.com*.

Bartolomé Palmer Riera es Ingeniero Técnico en Informática de Sistemas y actualmente está cursando Ingeniería Informática. Trabajó durante dos años como técnico informático en el *Consorci d'informàtica local* de Mallorca, realizando tareas de consultoría, helpdesk, y técnico de sistemas. Actualmente trabaja a tiempo parcial como consultor informático. *tolopalmer@gmail.com*

Diseño y simulación de un Filtro Pasivo de líneas de transmisión acopladas

Miguel Martinez Ledesma, Andre Luis Sousa Sena

Circuits d'Alta Freqüència (CAF)

Resumen— **Este documento especifica los resultados de diseño, implementación y simulación de un Filtro Pasivo de líneas de transmisión acopladas. Se trata de un filtro de alta frecuencia que utiliza el acoplamiento capacitivo e inductivo de varias líneas de transmisión para implementar un filtrado paso banda a la frecuencia de 2 GHz.**

I. INTRODUCCIÓN

Un filtro de microondas es una red de dos puertos (entrada y salida) utilizada para controlar la respuesta en frecuencia de un sistema, transmitiendo señal en una banda de paso y atenuándola en el resto de frecuencias. Cuando dos líneas de transmisión se sitúan una junto a la otra, la energía de viaja por una de ellas puede acoplarse en la otra a través de los campos electromagnéticos de la señal que viaja por la primera. Este tipo de líneas de transmisión son llamadas líneas de transmisión acopladas. Si estas líneas de transmisión acopladas se encuentran paralelas una a la otra y físicamente enterradas entre dos planos de masa y el substrato de una placa, se dice que utiliza una geometría Stripline. Pueden desarrollarse filtros de banda muy estrecha mediante el uso de un conjunto de N líneas de transmisión acopladas Stripline ajustadas para obtener un conjunto determinado de impedancias de línea complejas dependientes de la frecuencia, que tabuladas de forma correcta permiten ajustar con fidelidad cualquier tipo de filtro.

Para la implementación del filtro que deseamos obtener se han seguido las indicaciones de los capítulos 7º y 8º de [1], que especifican la formulación necesaria. Para precisar los valores puntuales que se deben utilizar en este diseño se han codificado un conjunto de algoritmos en lenguaje MATLAB y se ha verificado el correcto comportamiento simulando dichos parámetros mediante la versión estudiantil del software Ansoft Designer SV de la empresa Ansoft Corporation.

II. DISEÑO DEL FILTRO

El Filtro Pasivo de líneas de transmisión acopladas a diseñar está basado en el ejercicio 8.5 del Capitulo 8 de [1]. Se trata de un diseño de filtro paso banda con una respuesta con oscilaciones en la banda pasante de 0.5 dB, con un número de elementos igual a 3. La respuesta frecuencial de dicho filtro estará centrada en 2 GHz, y con un ancho de banda del 10% y una impedancia referencia de 50 Ohms (Figura 1).

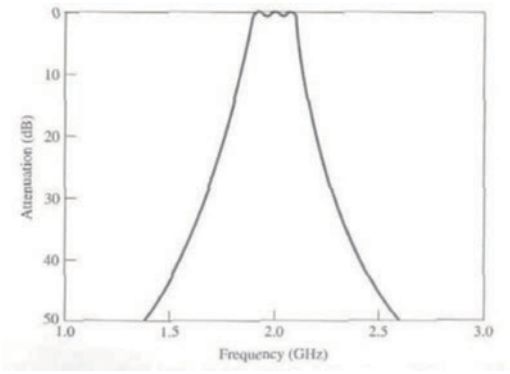


Fig. 1. Respuesta teórica del Filtro [1]

Para cumplir los requisitos de dicho filtro, se utilizan 3 líneas de transmisión acopladas (número de elementos N=3 en la Figura 2) ajustando sus características geométricas (longitud, anchura y separación entre pistas).

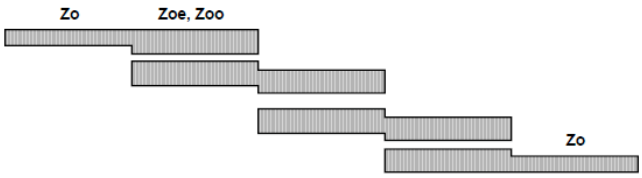


Fig. 2. Múltiples líneas de transmisión acopladas

Para ajustar la respuesta frecuencial de este diseño, nos basamos en la siguiente tabla donde se especifican los parámetros g_i (valores de los elementos de un filtro para filtros prototipos paso bajo normalizados a w_c=1) para 0.5 dB de ripple y 3 elementos (Tabla 1), obtenida de la Tabla 8.4 de [1].

TABLA III
VALORES PARA EL DISEÑO DE FILTROS PASO BAJO CON 0.5 dB DE RIPPLE

N	g_1	g_2	g_3	g_4	g_5	g_6
1	0.6986	1.0000				
2	1.4029	0.7071	1.9841			
3	1.5963	1.0967	1.5963	1.0000		
4	1.6703	1.1926	2.3661	0.8419	1.9841	
5	1.7058	1.2296	2.5408	1.2296	1.7058	1.0000

Utilizando los valores de la Tabla anterior podemos obtener los parámetros Zo_o y Zo_e (impedancia de la línea de transmisión en circuito abierto y en circuito cerrado) siguiendo la siguiente formulación:

$$\left.\begin{aligned} Z_{0e} &= Z_0\left[1 + JZ_0 + (JZ_0)^2\right] \\ Z_{0o} &= Z_0\left[1 - JZ_0 + (JZ_0)^2\right] \end{aligned}\right\} \quad (1)$$

Donde podemos calcular esos parámetros como:

$$Z_0 J_1 = \sqrt{\frac{\pi \Delta}{2 g_1}}$$

$$Z_0 J_n = \frac{\pi \Delta}{\sqrt{2 g_{n-1} g_n}} \; para \; n = 2,3,...,N \qquad (2)$$

$$Z_0 J_{N+1} = \sqrt{\frac{\pi \Delta}{2 g_N g_{N+1}}}$$

Siendo Δ el ancho de banda fraccional del filtro pasa banda, calculado mediante la siguiente formula:

$$\Delta = \frac{w_2 - w_1}{w_0} \qquad (3)$$

Donde w_0 es la frecuencia central del filtro paso banda, calculada como la media aritmética de w_1 y w_2, que son las frecuencias de corte de dicho filtro. Es decir, w_0 es la media aritmética siguiendo la siguiente formulación:

$$w_0 = \sqrt{w_2 w_1} \qquad (4)$$

En dichas formulas los parámetro J_i se corresponden con las constantes de las admitancias de los inversores del circuito equivalente de las líneas de transmisión acopladas utilizadas. Dicho esquema equivalente de las líneas acopladas puede observarse en la siguiente figura, donde θ = π/2.

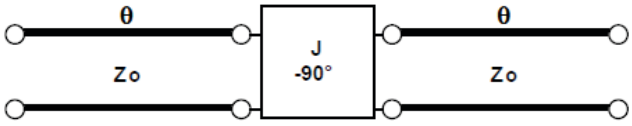


Fig. 3. Circuito equivalente de las líneas de transmisión acopladas [1]

En nuestro diseño las formulas anteriores dan como resultado según [1] los siguientes valores:

TABLA IV
VALORES PARA Z_{0O} Y Z_{OE} OBTENIDOS EN NUESTRO DISEÑO

n	g_n	$Z_0 J_n$	$Z_{0e}(\Omega)$	$Z_{0o}(\Omega)$
1	1.5963	0.3137	70.61	39.24
2	1.0967	0.1187	56.64	44.77
3	1.5963	0.1187	56.64	44.77
4	1.0000	0.3137	70.61	39.24

A la hora de implementar nuestro diseño como líneas de transmisión acopladas seleccionamos la tecnología Stripline, ya que en el capitulo 7 de [1] podemos encontrar la formulación necesaria para encontrar los parámetros geométricos de diseño b (anchura de la Stripline), ε_r (permitividad del material), S (separación entre líneas) y W (anchura de líneas) de la siguiente figura.

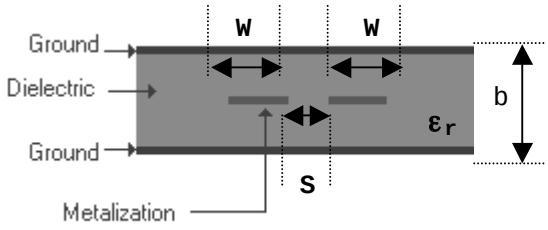


Fig 4. Visualización de los parámetros geométricos en líneas acopladas Stripline

Sabiendo que la longitud de las pistas a construir en el filtro se corresponde con λ/4 (que se corresponde con θ = π/2 en la ecuación 8.102 de [1]) con respecto a la frecuencia de la señal, donde es la λ longitud de onda de la frecuencia deseada en el medio de transmisión, por lo tanto (siendo λ_0 la longitud de onda en el vacío, *c* la velocidad de la luz en el vacío, y ε_r la constante dieléctrica del material – siguiendo el ejercicio 3.5 del capítulo 3.7 de Striplines de [1]) podemos aproximarla por:

$$L = \frac{\lambda}{4} = \frac{\lambda_0 / \sqrt{\varepsilon_r}}{4} = \frac{c / (f\sqrt{\varepsilon_r})}{4} \qquad (5)$$

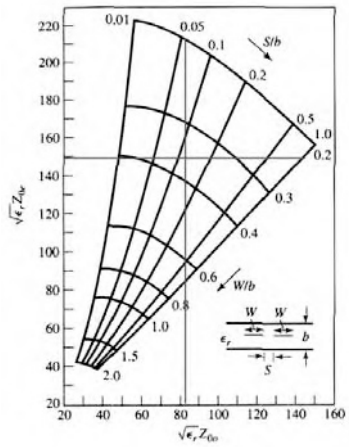


Fig. 5. Gráficas de S y W en función de Zo_o y Zo_e [1]

Los parámetros W (anchura de la pista) y S (separación entre pistas) se calculan utilizando la siguiente formulación (del capitulo 7 de [1]):

$$Z_{0e} = Z_0 \frac{b^2 - S^2}{4bW\sqrt{\varepsilon_r}}$$

$$Z_{0o} = Z_0 \frac{1}{2W\sqrt{\varepsilon_r}\left[\frac{2b}{b^2 - S^2} + \frac{1}{S}\right]} \qquad (6)$$

El capítulo 7º de [1] incluye además la siguiente gráfica (Fig.5) para el cálculo aproximado de los parámetros S y W en función de Zo_o y Zo_e.

III. ALGORITMOS IMPLEMENTADOS

Con la intención de obtener una mayor precisión de los parámetros S y W, en lugar de calcularlos de forma gráfica en las figuras de [1], se han implementado un conjunto de códigos en lenguaje MATLAB.

Se ha implementado un algoritmo que, utilizando las ecuaciones (1) y (2), calcula los parámetros Zo_o y Zo_e para un número de 3 elementos y los parámetros g_i de la Tabla I (con N=3). El resultado de dicho algoritmo es:

```
-------------Level n = 1-------------
JZ = 0.313691
Zoe = 70.604689 Zoo = 39.235544
-------------Level n = 2-------------
JZ = 0.118719
Zoe = 56.640634 Zoo = 44.768776
-------------Level n = 3-------------
JZ = 0.118719
Zoe = 56.640634 Zoo = 44.768776
-------------Level n = 4-------------
JZ = 0.313691
Zoe = 70.604689 Zoo = 39.235544
```

Se puede comprobar que los parámetros son los obtenidos en la Tabla II, pero con mayor resolución.

El segundo algoritmo implementado calcula los valores de S y W de líneas acopladas, minimizando los valores de dichos parámetros para obtener los parámetros Zo_o y Zo_e deseados. Se basa en la implementación de las ecuaciones de (6), y dados un conjunto Zo_o y Zo_e, minimiza el error de cálculo de los coeficientes S y W por aproximación lineal del punto de trabajo. Para hacer esa tarea se han dado valores a Z_{0o} y Z_{0e} en las ecuaciones (6) y éstas se han igualado a 0. El algoritmo utiliza dichas reducciones para encontrar los puntos de trabajo dando valores de S y W para encontrar los valores que igualen dichas funciones a cero.

Las figuras 6.a y 6.b representan las gráficas de minimización de las funciones (6) para obtener los puntos S y W:

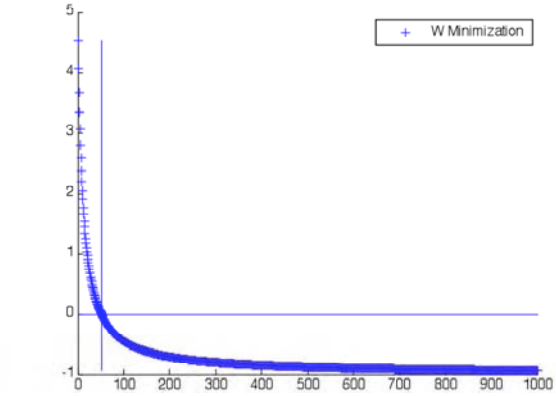


Fig. 6.a. Gráfica de obtención de W.

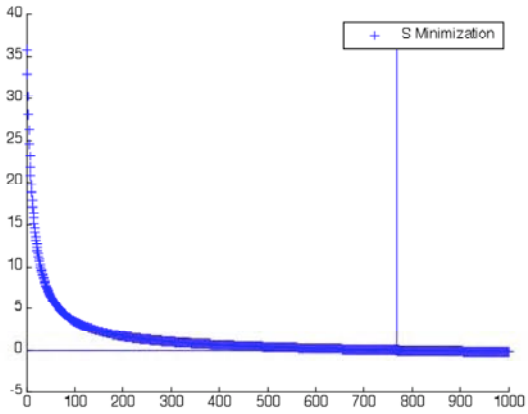


Fig. 6.b. Gráfica de obtención de S.

En la siguiente figura puede observarse la posición de uno de los puntos obtenidos sobre las curvas de distintos valores de S y W, utilizando el algoritmo anterior.

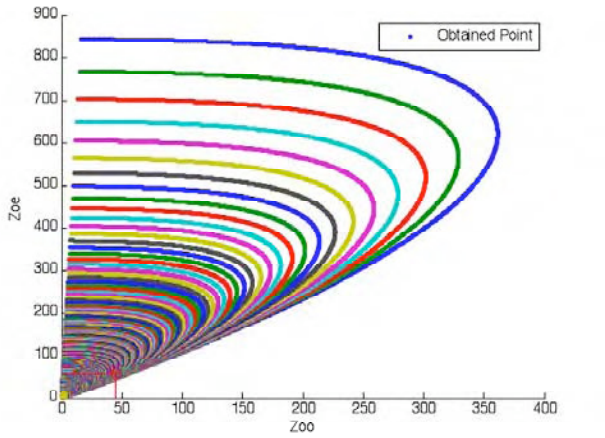


Fig. 7. Visualización de los puntos obtenidos

IV. SIMULACION DEL FILTRO

Utilizando el software Ansoft Designer SV, y siguiendo los pasos del manual de dicho software para la implementación de filtros acoplados [2], se ha simulado el comportamiento de los parámetros obtenidos mediante los algoritmos implementados.

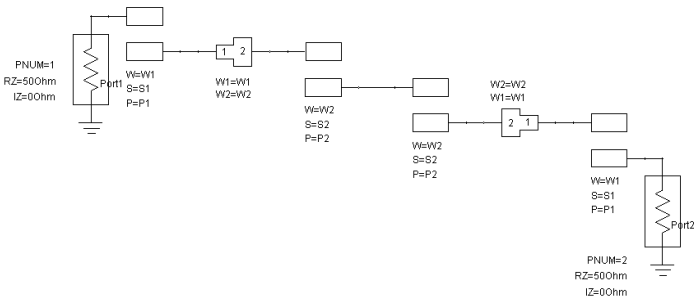


Fig. 8. Esquema del circuito implementado.

El primer paso para la simulación es la selección del tipo de substrato utilizado. En nuestro caso seleccionamos el tipo de placa FR4 con ε_r = 4.4 y b = 60 mils. Posteriormente implementamos el circuito basado en

Striplines indicado en [2]. Este circuito puede observarse en la Figura 8.

En este diseño se han implementado 3 líneas acopladas y 2 acopladores de línea. Tanto las líneas como los acopladores tienen los siguientes parámetros:

TABLA V
PARÁMETROS INTRODUCIDOS EN LA SIMULACION

Parámetro	Valor
S1	0.0003494469
W1	9.86842105e-5
S2	0.0005592105
W2	6.53127383e-5
P1	0.017877424
P2	0.017877424

Donde Si y Wi han sido obtenidos de los algoritmos anteriormente desarrollados, y Pi corresponde a la longitud de las líneas.

V. RESULTADOS DE LA SIMULACION

Una vez implementado el circuito e introducidos los parámetros anteriores, se implementó un análisis en frecuencia entre 0.5 y 3.5 GHz con un step de 50 MHz de los parámetros S_{11} y S_{12} del circuito. El resultado de esta simulación puede observarse en la siguiente figura, donde el parámetro S_{11} está representado en azul, y el parámetro S_{12} en rojo.

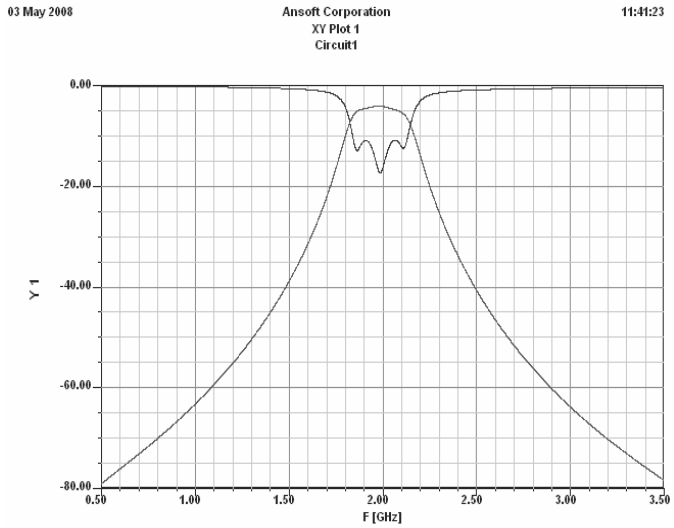


Fig. 13. Análisis de los parámetros S_{11} y S_{12}

Como puede observarse, el parámetro S_{12} (transferencia que sufre la entrada a la salida del circuito) tiene un comportamiento de filtro paso banda a la frecuencia central de 1.99 GHz, con un ancho de banda de 200 MHz aproximadamente (1.84GHz a 2.14GHz). El hecho de no tener la frecuencia central posicionada exactamente en 2 GHz está relacionado con la precisión numérica utilizada, así como la resolución en el análisis de los parámetros de transferencia.

Puede observarse que a su vez el parámetro S11 (rebote de la señal de entrada en la misma entrada) sufre un desvanecimiento que es máximo en la frecuencia central del filtro (y aproximadamente igual a -20 dB).

Este efecto está relacionado con el hecho de que la potencia de la señal se transfiere a la salida en la frecuencia de trabajo, es decir, la entrada del sistema está adaptada a la salida si trabaja a dicha frecuencia, mientras que si la frecuencia a la entrada varía, el porcentaje de la señal transferida a la salida se minimiza.

VI. CONCLUSIONES

Se ha conseguido implementar el diseño propuesto con las
características requeridas de forma correcta. Además, se han implementado un conjunto de herramientas que han permitido mejorar la resolución de cálculo de los parámetros internos del filtro con respecto a los obtenidos en [1].

REFERENCIAS

[17] David M. Pozar, *Microwave Engineering*, 2nd ed., John Wiley & Sons Inc., 1998.
[18] Getting Started with Ansoft Designer, rev. 1.1, Ansoft Corporation, 2003.

ASSIGNATURA IMPARTIDA PER
Bartomeu Alorda y Eugeni Garcia

Miguel Martínez Ledesma
Diplomado en Ingeniería Técnica de Telecomunicación, especialidad en Telemática,
Universitat de les Illes Balears (UIB), 2003.
Palma de Mallorca – España.
Trabajo actual: IMEDEA CSIC-UIB
Área de interés: Tecnología Submarina.
Cursando: Master Ingeniería Electrónica - UIB

André Luis Sousa Sena
Licenciado en Ingeniería Eléctrica, especialidad en Telecomunicación.
Faculdade de Ciência e Tecnologia (Área 1),2005.
Salvador - Bahia - Brasil.
Trabajo actual: Becario (FPI-GTE-UIB)
Área de interés: Tecnología Submarina
Cursando: Master Ingeniería Electrónica - UIB

Desenvolupament a baix nivell per aplicacions en Temps Real

MAS BONED, F. B., OLIVER TORRES, T.

Resum— **El present article donarà a conèixer les diferents aplicacions a baix nivell en temps real que podem trobar a internet, centrant el seu contingut a les llibreries C++ i les aplicacions RTP tools (desenvolupades pel doctor Henning Schulzrinne i el seu departament). El programari que ens ofereix Sun Microsystems per dur a terme les mateixes accions (en aquest cas emprant el llenguatge de programació Java), i les canonades de GStreamer per aplicacions de capes superiors es comentaran molt breument. El protocol més implicat dins la transferència en temps real seran breument explicat.**

I. Introducció

La transmissió d'informació multimèdia (àudio i vídeo) té un gran impacte en els sistemes de comunicació. Aquesta transmissió requereix no tan sols el suport de les xarxes, sinó també dels protocols de comunicació de nivells superiors. Aquest protocols són els adoptats per la IETF: RTP, SDP i SIP.

Podem parlar de transmissió d'informació en temps real quan es pot assegurar que la informació arriba al seu destí amb uns paràmetres determinats (retràs, rendiment, fiabilitat, etc.).

En general les aplicacions multimèdia requereixen una qualitat de servei (QoS) per part dels serveis de la xarxa. De les noves xarxes s'exigeix poder especificar aquesta qualitat de servei i assegurar el seu compliment.

Dissenyats els protocols i les xarxes capaces de mantenir un transferència en temps real, només necessitem les llibreries que ens permetin la creació d'una aplicació (programari) d'aquestes característiques. Així neixen les llibreries C, C++ i Java, entre d'altres.

II. RTP

Les aplicacions multimèdia necessiten alguna classe de protocol de transport, amb característiques diferents de TCP i amb major funcionalitat que UDP. El protocol creat per satisfer aquestes necessitats és l'anomenat 'Real-time Transport Protocol' (RTP). No obstant RTP es considera un protocol de capa d'aplicació.

Les seves funcions són les següents:

- Comunicar la elecció de l'esquema de codificació de les dades.
- Determinar la relació temporal entre les dades rebudes.
- Sincronitzar els distints medis.
- Indicar la pèrdua de paquets.
- Indicar límits de *frames* en les dades.
- Identificació amigable d'usuaris.

Un requeriment addicional a les funcions senyalades és l'ús eficient de l'ample de banda.

En realitat l'estàndard RTP defineix dos protocols, RTP i RTCP (Real-time Transport Control Protocol). El primer es emprat per l'intercanvi de dades multimèdia, mentre que el segon és emprat per l'enviament periòdic d'informació de control associada a un determinat flux de dades.

I. Format del paquet RTP

El format que segueixen les trames RTP és el representat a la figura 1:

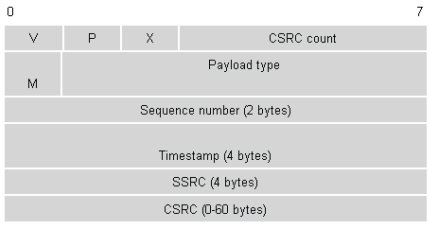


Figura 1. Capçalera del paquet RTP.

On 'V' són dos bits que ens informen de la versió del protocol. El bit 'P' ens indica si ha hagut farciment de bits per complir algun requeriment mínim. El bit 'X' senyala la presència d'extensions de l'encapçalament. El bit de marques 'M' s'empra per indicar *frames*.

J. RTCP

RTP lliura un *stream* de control que està associat a un *stream* de dades per una aplicació multimèdia.

Proveeix tres funcions:

- Feedback en el desenvolupament de l'aplicació.
- Sincronització i correlació de diferents *streams* que provenen de la mateixa font.
- Un mode de transportar la identitat de l'usuari emissor pel desplegament en la interfície de l'usuari receptor.

Un cop esmentat el protocol RTP descriurem les eines de desenvolupament a baix nivell en Temps Real per a diferents llenguatges de programació.

III. RTP tools

Les RTP tools són eines que ens proporciona la Universitat de Columbia, creades pel doctor Henning Schulzrinne i els seus col·laboradors [1]. Aquestes aplicacions estan codificades en C, per tant poden ser visualitzades per qualsevol compilador C.

K. Descripció

La distribució de les RTP tools es composa d'una sèrie de petites aplicacions que poden ser emprades per el processament de dades RTP.

- rtpplay: grava la sessió RTP enregistrada per rtpdump.

- rtpsend: genera paquet RTP amb la descripció textual.
- rtpdump: analitza sintàcticament i imprimeix els paquets RTP.
- rtptrans: traductor RTP entre xarxes *multicast* i *unicast*

L. Instal·lació

Les RTP tools han de poder ser compilades sobre qualsevol plataforma Posix compatible amb els sockets de suport.

M. Suggeriment d'ús general

Les adreces de xarxa poden ser adreces *multicast* o *unicast*. Aquestes poden ser especificades amb nombres decimals o com el nom del host. El nombre del port s'ha d'especificar amb un nombre decimal dins el rang 1..65535.

A no ser que s'indiqui el contrari, les entrades arribaran per *stdin* i les sortides per *stdout* (fitxers d'entrada i de sortida respectivament).

N. rtpplay

rtpplay llegeix la sessió RTP de dades, emmagatzemada per rtpdump del fitxer 'f' o *stdin*.

5) Format de trama

rtpplay[-T][-v][-f][-p][-s][-b][-e][/ttl]
On 'T' i 'v' són banderes.
'f' és el fitxer d'entrada.
'p' és el fitxer del perfil de la freqüència del rellotge.
's' és el port font.
'b' i 'e' són el temps d'inici i final respectivament, mesurats en segons.

6) Funcionament

Si el fitxer no està especificat, s'enviaran les dades a l'adreça de xarxa de destí i port amb un temps de vida de 'ttl'.

Si la bandera 'T' està activada, el temps entre paquets correspon al temps d'arribada en lloc del *timestamps* de RTP. Per una altra banda podem emprar el *timestamps* per suavitzar el *jitter* i restaurar la seqüència del paquets. Aquest fet pot provocar un desordre en el paquets de RTP i RTCP.

El port font és el port de sortida dels paquets, es pot establir amb la bandera 's'. S'escull un port a l'atzar si no activem la bandera.

El fitxer del perfil conté línies amb dos camps cada una: la primera és la carrega útil de tipus numèric, a segona és la freqüència del rellotge. En cas de que la bandera 'T' estigui desactivada els perfils seran ignorats.

La bandera 'v' ens indica que s'han generat paquets a *stdout*.

O. rtpsend

L'aplicació rtpsend envia un flux de paquets RTP amb paràmetres configurables. Amb això s'intenta posar a prova les característiques del protocol RTP. Les capçaleres de RTP i RTCP són llegides d'un fitxer, generat a mà, per un programa o per rtpdump (format ascii).

1) Format de trama

rtpsend [-a][-l][-s][-f]destí/port[/ttl]
On 'a', 'l' i 's' són banderes.
'f' és el fitxer descriptiu.
'ttl' és el temps de vida.

2) Funcionament

Els paquets se envien amb un temps de vida de valor 'ttl'.

Si les dades es llegeixen d'un arxiu en lloc de *stdin*, la bandera 'l' reenvia la mateixa seqüència de paquets una i una altra vegada.

El port d'origen per els paquets de sortida es pot configurar amb la bandera 's'. Un port a l'atzar s'escull si no s'especifica aquest indicador.

Si la bandera 'a' s'especifica, rtpsend inclou un *router* IP opció d'alertar els paquets RTCP. Això s'empra per els recursos YESSIR de reserva de protocol.

L'arxiu conté la descripció dels paquets que s'enviaran. Dins l'arxiu, cada entrada comença amb un valor de temps, en segon, en relació amb el temps de començament de la trama. El valor temporal ha d'aparèixer a l'inici d'una línia, sense espais en blanc. Dins d'un RTP, RTCP o descripció de paquets, els paràmetres poden aparèixer en qualsevol ordre, sense espais en blanc al voltant del signe igual.

Les línies se continuen amb els espais en blanc inicials en la línia següent.

Les línies comencen amb el signe #. Els *strings* s'inclouen entre cometes.

P. rtpdump

Analitza els paquets RTP per poder ser enviats posteriorment.

1) Format

rtpdump [-F][-t][-x][-f][-o]
On 'F' és el format.
't' és la duració.
'x' són les dades (*payload*).
'f' és l'arxiu d'especificació d'adreça de xarxa.
'o' és el fitxer de sortida.

2) Funcionament

L'aplicació rtpdump escolta l'adreça i el port del parell de paquets RTP i RTCP i descarrega la versió al fitxer de sortida si s'especifica o al *stdout*.

Si el fitxer 'f' s'especifica, el fitxer s'empra en lloc de l'adreça de xarxa.

Format	**Text/binari**	**Format**	**Text/binari**
Dump	binari	Ascii	text
Header		Hex	
Payload		rtcp	
		short	

TAULA 1. FORMAT DE LES TRAMES.

Q. rtptrans

Els paquets RTP/RTCP arriben de una de les direccions a totes les direccions.

Les direccions poden ser *multicast* o *unicast*. El valor 'ttl' per les direccions *unicast* s'ignoren (en realitat, no comprova si són paquets RTP o no).

1) Format de trama

rtptrans [host]/port[/ttl][host]/port[/ttl][...]

On 'ttl' és el temps de vida.

2) Funcionament

Amés el traductor pot traduir paquets VAT en paquets RTP. El control de VAT es tradueix en paquets RTCP SDES amb n CNAME i un nom d'entrada. No obstant, això només està destinat a ser emprat en les següents configuracions: paquets VAT que arriben a una connexió *multicast* es tradueixen en RTP i es transmeten a través d'una connexió *unicast*.

Per lo tant, en la actualitat es compte principalment amb el suport de les següents tipologies: *multicast* VAT→ Traductor→ *unicast* RTP, i en el camí de tornada *multicast* VAT ← traductor ← *unicast* VAT.

Això significa que l'agent d'àudio i vídeo en l'enllaç *unicast* ha de poder emprar tant VAT com RTP.

IV. GNU ccRTP

ccRTP és una llibreria de C++ basades en GNU Common C++ [2], que proporciona un alt rendiment, flexibilitat i extensibilitat amb els estàndards de a pila de protocols RTP i amb total suport de RTCP. El disseny i l'aplicació de ccRTP és adequat per servidors d'alta capacitat i passarel·les, així com per aplicacions d'usuaris.

La llibreria està creada com un protocol de *framework* i no com un protocol de transport típic tal com TCP o UDP. Així, RTP es casi mai implementable fora de la capa d'aplicació. Les aplicacions RTP moltes vegades han de personalitzar la capacitat d'adaptació del disseny i processament dels paquets RTP, les limitacions del temps, període de sessions, així com altres mecanismes de RTP i RTCP.

El suport a les dades d'àudio i vídeo també es consideren a les llibreries ccRTP, que pot fer marcs parcials (ensamblatge i desensamblatge). Suporta el models de transport *unicast*, *multicast* i *multi-unicast*, així com múltiples fonts actives sincronitzades, múltiples sessions RTP i múltiples aplicacions RTP. Això permet el seu ús per a la construcció de totes les formes d'internet basades en les normes d'àudio i vídeo per sistemes de conferència.

ccRTP proporciona seguretat i alt rendiment. Empra la cua de les llistes de paquets per la repetició i transmissió de paquets de dades. GNU ccRTP dóna suport a RTCP i molt d'altres estàndards que es necessiten per avançades aplicacions d'*striming*.

3) Situació actual

GNU ccRTP 1.5.0 introdueix suport per el perfil Secure RTP.

Les emissions 1.4.x s'han centrat en la introducció del protocol IPv6

4) Manual de referència

Aquest petit manual defineix les classes C++ de les llibreries de ccRTP per desenvolupar una comunicació en temps real.

a) Iniciar sessió:

Iniciar sessió RTP amb ccRTP implica la construcció d'un objecte de la classe i crida alguns mètodes d'inicialització opcionals de *RTPSession*. El propòsit de la pila de protocols de ccRTP, com *RTPSession*, ha de ser signar abans de començar l'execució. Això es fa cridant el mètode *startRunning()*. També hi ha un *enableStack()*, aquest tan sols activa la pila, però no s'inicia l'execució.

El període de sessions i altres paràmetres es pot establir com:

- Afegir destinataris.
- Establir format de càrrega útil.
- Conjunt local d'articles SDES.
- Permetre el farciment de paquets de dades.
- Establiment de les direccions.
- Establir l'ample de banda cridant el mètode *setSessionBandwidth()*. De no ser així, tindrem per defecte un ample de 64 Kbytes, i un 5% d'aquest per RTCP.

b) Enviament de dades:

El paquets de dades s'envien a través del mètode *putData()*.

Per defect, el bit marcador de paquets enviats no està definit. El seu valor es pot configurar a través del mètode *setMark()*, que té un *boolean* com argument.

ccRTP també suporta fragmentació de blocs de dades en varis paquets RTP. Quan els blocs de dades major que el màxim tamany del segment es presenta a través de *putData()*, dos o més paquets s'introdueixen en la cua de paquets sortints.

c) Recepció de dades:

Rebudes les dades s'extreu el paquet de la cua d'entrada a través del mètode *getData()*, el qual ens dóna el *timestamp* i una font opcional de sincronització.

El nou paquet de cues s'encarrega de funcions tal com reordenació de paquets o filtrat de paquets duplicats.

Es pot saber si hi ha paquets en la cua de recepció mitjançant *isWaiting()*. Ambdós, getData() i isWaiting(), tenen un paràmetre opcional el qual selecciona una determinada font de sincronització de la unitat de dades. Si no s'especifica les unitat de dades es retornen, independentment del seu origen.

El períodes de les sessions de RTP poden tancar suprimint o destruint els objectes *RTPSession*. La pila envia un paquet BYE a cada destí quan el destructor de sessions es cridat. També es possible explícitament enviar un paquet BYE a través del mètode *dispatchBYE()*.

d) Tipus de càrrega útil i formats:

En el context de RTP, una càrrega útil a RTP és un identificador numèric de set bits. Per tipus de càrrega útil, GNU ccRTP defineix el tipus enter *PayloadType* i el tipus enumerat *StaticPayloadType*, com l'enumeració de la càrrega útil de tipus de RTP assignats per àudio i els formats de vídeo estàndard.

e) Manipulació de successos:

Hi ha una sèrie de casos que exigeixen una especial resposta de la sol·licitud. ccRTP defineix *plug-ins* per manejar aquests successos.

- Paquets RTP d'arribada:
  - *onRTPPacketRecv*
- Paquets RTCP d'arribada:
  - *onGotSR*
  - *onGotRR onGotRR*
  - *onGotSDESChunk*
  - *onGotAPP onGotAPP*
  - *onGotRRSRExtension*
- Sincronització d'estats font:
  - *onNewSyncSource*
- Col·lisions SSRC:
  - *onSSRCCollision*
- Paquets de caducitat de RTP
  - *onExpireSend*
  - *onExpireRecv*
  - *end2EndDelayed end2EndDelayed*

V. Llibreries Java

JMF és una API que ens serveix per incloure multimèdia a les nostres aplicacions de Java [3], funciona bàsicament, rebent el contingut multimèdia d'alguna font, per després processar-lo i entregant-lo en alguna sortida multimèdia.

JMF es construeix entorn a una arquitectura de components: mitjans manipuladors, font de dades, codecs, renderers i mux/demux.

Exemples de mètodes: start(), stop(), realize(), prefetch(), deallocate(), setRate(), setMediaTime(), getVisualComponent(), getControlPanelComponent(), etc.

JMF és una molt flexible arquitectura multimèdia que ens mostra moltes promeses. En el futur, cal esperar que Sun treballarà per fer-ho més estable, així com documentar el marc i proporcionant un major nombre de codis exemple.

VI. Canonades GStreamer

El soport de RTP en Gstreamer [4] actualment és parcila, encara així, ja és possible executar algunes canonades que ens transmeten vídeo i àudio empratn RTP damunt UDP.

Per poder emprar aquestes funcions cal tenir instal·lats els següents *plugins*: gstreamer0.10-plugins-base, gstreamer0.10-farsight, gstreamer0.10-plugins-good, gstreamer0.10-plugins-ugly, gstreamer0.10-alsa, gstreamer0.10-ffmpeg, gstreamer0.10-x.

Les funcions que en permeten fer la transferència són:

- rtpbin: s'adapta a qualsevol necessitat.
- rtpjitterbuffer: control del jitter i ordena els paquets.
- Rtpdemux: detecta qualsevol nou tipus de càrrega útil que es rebi en el flux RTP.

VII. Conclusions

La creació d'aquest article ens ha fet veure la difícil tasca que tenen els investigadors. La redacció de l'article implica un gran aprofundiment dins el tema o temes esmentats a aquest. Si bé amb menys mesura que els investigadors de la Universitat de les Illes Balears, nosaltres em procurat capficar-nos i extreure la major informació possible sobre la transferència en temps real.

Referent a l'article, cal esmentar que la transferència en temps real per dos usuaris qualsevol de la xarxa, emprant programari propi té una certa dificultat. Els coneixement en llenguatge C++, C i Java han de ser elevats per tal d'aconseguir un transferència d'informació amb unes condicions acceptables.

Les canonades GStreamer es poden utilitzar una vegada aconseguida la transferència i fer la reproducció de la informació enviada.

En definitiva, podem concloure que la creació de l'article ha fomentat el nostre esperit crític i que la codificació d'un programa, mitjançant programari lliure, necessita evolucionar per tal d'oferir els mateixos i millors serveis, de forma més senzilla que la esmentada a aquest article.

Referències

[19] RTP tools. Disponible on-line: http://www.cs.columbia.edu/~hgs/
[20] GNU ccRTP. Disponible on-line: http://www.gnu.org/software/ccrtp/
[21] Java media framework. Disponible on-line: http://java.sun.com/javase/technologies/desktop/media/jmf/
[4] Canonades GStreamer. Disponible on-line: http://crysol.inf-cr.uclm.es/node/655

Nom: Francisco de Borja Mas Boned.
Titulació: Enginyeria Tècnica en Telecomunicacions, especialitat en Telemàtica.
Direcció: fborja.mas@gmail.com

Nom: Tomeu Oliver Torres.
Titulació: Enginyeria Tècnica Industrial, especialitat en electrònica industrial.
Enginyeria Tècnica en Telecomunicacions, especialitat en Telemàtica.
Direcció: tomeu.oliver@gmail.com

Assignatura impartida per: Magdalena Payeras Capellà

Análisis de la respuesta frecuencial de un filtro pasa-banda mediante Labview (Instrumentación GPIB)

Eduardo Azcona Soria, Jorge Torres Lobera, Pedro Rodríguez Riquero

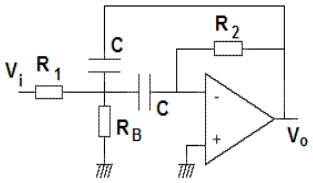
Asignatura: Instrumentació Electrònica II

A lo largo del documento veremos como se puede obtener la respuesta en frecuencia de un filtro pasa-banda. Se explicará el programa principal, los subprogramas creados para facilitar la estructuración y una serie de pruebas gráficas.

VIII. DISEÑO DEL FILTRO

En primer lugar y antes de presentar nuestro filtro y sus características más importantes, debemos recordar que un filtro electrónico es un circuito que altera la amplitud y la fase de una señal de entrada en función de su frecuencia para poder amplificar o atenuar señales dentro de un determinado rango de frecuencias.

En nuestro caso decidimos utilizar un filtro activo que permiten ajustar la ganancia con un valor superior a la unidad. A continuación, presentaremos el circuito y la función de transferencia obtenida (Ec.1):



$$H(s)=\frac{V_O}{V_I}=\frac{K\cdot\left(\frac{\omega_o}{Q}\right)\cdot s}{s^2+\left(\frac{\omega_o}{Q}\right)\cdot s+\omega_o^2}$$

Fig. 1.1 Filtro pasa banda activo

$$H(s)=\frac{V_O}{V_I}=\frac{-\left(\frac{1}{R_1\cdot C}\right)\cdot s}{s^2+\frac{2}{R_2\cdot C}\cdot s+\frac{R_1+R_B}{R_1\cdot R_2\cdot R_B\cdot C^2}} \qquad \textbf{Ec. 1}$$

Como podemos observar, podremos determinar los distintos parámetros que definen nuestro filtro pasa-bandas: ganancia K del filtro, frecuencia de corte ω_o, factor de calidad del filtro Q y ancho de banda del filtro BW. Una vez obtenidas las expresiones y asignados valores a dichos parámetros, podremos aislar el valor de los componentes.

$$\omega_O=2{,}0788\cdot10^4\,\frac{rad}{s}\qquad f_O=3308Hz$$

$$Q=5\quad BW=636{,}62Hz\quad K=3\quad C=15nF$$

$$R_1=-\frac{Q}{\omega_O\cdot K\cdot C}\approx5{,}6k\Omega\qquad R_2=\frac{2\cdot Q}{\omega_O\cdot C}\approx33k\Omega \qquad \textbf{Ec. 3}$$

$$R_B=\frac{Q}{\omega_O\cdot C\cdot\left(2\cdot Q^2+K\right)}\approx314\Omega$$

Una vez obtenidas todas las expresiones, el funcionamiento ideal del circuito se corresponderá con la figura 1.2. Cuanto mayor sea la calidad del filtro, podremos obtener un rango más restringido de frecuencias. En nuestro caso, el factor de calidad es igual a 5, puesto que no se pretende implementar un filtro excesivamente selectivo en ancho de banda.

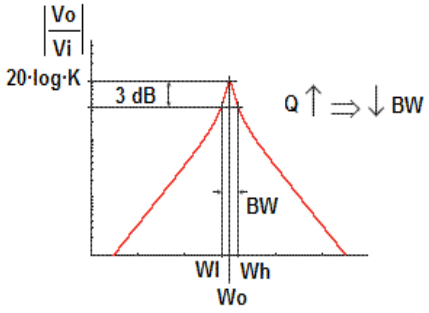


Fig. 1.2 Representación del filtro pasa banda

Antes de comenzar a describir cada uno de los indicadores y controles existentes en nuestro panel frontal así como el propio programa, debemos decir que el entorno de programación de LabVIEW se basa en la programación gráfica o lenguaje G; debido a ello, los indicadores y controles que introduzcamos en el panel frontal nos permitirán poder observar el valor de variables, gráficas, indicadores o controles de estado mientras se ejecuta el programa.

IX. DESCRIPCIÓN GENERAL DEL PROGRAMA

El objetivo del programa es el de proporcionar información relevante sobre las características de un filtro pasa banda. En particular es capaz de realizar las siguientes funciones:

- Gráfica con el módulo de la respuesta frecuencial
- Gráfica con la fase de la respuesta frecuencial
- Ancho de banda
- Frecuencia de resonancia
- Factor de calidad
- Generación de un fichero con los datos obtenidos en el módulo de la respuesta frecuencial
- Selección entre escala logarítmica o lineal
- Selección de la frecuencia inicial, final y número de pasos del barrido frecuencial
- Selección de la amplitud de la onda de entrada

Para implementar todos los puntos se ha creado un VI principal separado por un bloque “sequence” que lo divide en cuatro partes. Cada una de ellas se encuentra claramente diferenciada por el conjunto de acciones que incorpora, tal y como se puede ver en la Figura 2.1.

Fig. 2.1 Estructura del programa principal

La primera parte se encarga de inicializar todos los instrumentos a utilizar (Generador de funciones, multímetro y osciloscopio).

El segundo paso se caracteriza por incorporar un bloque "for", en el cual se realiza el barrido de frecuencias, el cálculo de la fase y la medida de la magnitud. Al finalizar las iteraciones, que coinciden con el número de pasos a muestrear, tendremos tres matrices, una con las frecuencias, otra con la magnitud y la última con la fase. Para implementar estas funcionalidades, se han creado tres subvi, uno que calcula el desfase temporal para cada frecuencia, otro encargado de controlar el generador de funciones y un tercero cuya función se reduce a controlar la adquisición de la amplitud de la onda de salida, como veremos más adelante. Finalmente, se grafican los conjuntos de pasos obtenidos y se obtiene la frecuencia angular central (*fo*).

En la tercera parte se cierran los instrumentos que ya no utilizaremos (multímetro y generador) y se preparan los cálculos para obtener el ancho de banda. Para ello, creamos una estructura iterativa, que se ejecuta tantas veces como número de pasos contenga el barrido frecuencial, recorriendo las amplitudes hasta obtener "*fo-3db*", es decir, "*fh*" y "*fl*".

En último paso se calcula el ancho de banda y el factor de calidad, con los valores recibidos anteriormente. Finalmente, se genera un fichero con los datos obtenidos en el módulo de la respuesta frecuencial, del paso dos. Para ello, se abre un fichero, se escribe en él mediante una estructura iterativa y se cierra.

Estos cuatro pasos conforman la totalidad del programa y una vez ejecutados, se vuelve a repetir el ciclo tantas veces como queramos.

X. Descripción del panel frontal

Por lo que respecta a nuestro panel frontal, podríamos dividirlo en cuatro partes:

En primer lugar, disponemos de una serie de controles numéricos, tanto de pestaña como de rueda, que nos permiten controlar: la amplitud de la onda de entrada, el número de pasos a evaluar y la frecuencia inicial y final del análisis. También disponemos de un indicador que nos permite saber para qué frecuencia se está ejecutando el programa.

En segundo lugar, disponemos de dos gráficos en los cuales representaremos la magnitud y la fase. También disponemos de un interruptor, que nos permite determinar la escala de frecuencia de la magnitud; logarítmica o lineal.

En tercer lugar, disponemos de una serie de controles de pestaña que nos permiten controlar la acción a realizar: leer o escribir, sobre un archivo determinado por otro controlador que nos permite introducir el 'path' de dicho archivo.

En cuarto lugar, disponemos de cinco indicadores numéricos: frecuencia de resonancia, ancho de banda, frecuencia inferior y superior del ancho de banda y factor de calidad. La figura 3.1 muestra nuestro panel frontal.

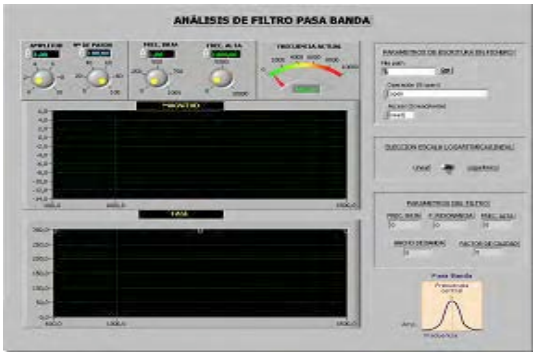


Fig. 3.1 Panel frontal

XI. Control del multímetro y generador de funciones (SUBVI)

El control de los instrumentos viene facilitado por las librerías que nos proporcionan los fabricantes.

En el caso del generador, una vez inicializado, se habilita la salida y se permite su control, introduciendo los parámetros de la onda a generar.

Fig. 4.1 Bloque del generador de funciones

El multímetro, después de inicializarse y especificarse el tipo de señal a leer, en nuestro caso AC, se limita a proporcionar la medida en cuestión.

Fig. 4.2 Bloque de lectura del multímetro

XII. Obtención de la fase de la respuesta frecuencial

Con la obtención de la respuesta en magnitud y fase del circuito a analizar, disponemos de información suficiente como para decidir si será útil en nuestro sistema final. Este estudio es imprescindible para elegir los filtros y amplificadores a incorporar.

R. Planificación

Al la hora de estudiar la respuesta frecuencial de un filtro, es esencial conocer su respuesta en magnitud, es decir, la atenuación que sufre a lo largo del barrido frecuencial, no obstante, en muchas ocasiones también puede sernos de gran utilidad la fase. De este modo obtendremos el desfase de la onda de salida respecto a la de entrada, o lo que es lo mismo, el desplazamiento temporal que sufre la salida respecto a la entrada.

Para realizar un estudio de la fase mediante Labview nos haremos valer de la siguiente expresión:

$$Fase(^{o}) = 360^{o} \cdot dT(seg) \cdot frec(Hz) \qquad \textbf{Ec.4}$$

La ecuación 4 es una relación a partir de la cual obtener la respuesta en fase para una determinada frecuencia. Si

analizamos detenidamente la expresión, nos guiará a la hora de planificar el proceso de implementación. Vemos esquemáticamente los puntos a seguir:

1. Configurar mediante Labview los canales del osciloscopio.
2. Adquirir la onda de salida y entrada mediante un osciloscopio de dos canales.
3. Transmitir las señales capturadas hacia el ordenador para su posterior procesamiento.
4. Determinar los pasos por cero de la señal.
5. Comprobar que los pasos por cero hayan sido iguales en las dos señales (de positivo a negativo o viceversa).
6. Asegurarse que en ningún caso el desfase temporal sea mayor a un periodo.
7. Aplicar la ecuación 1.
8. Cuando tenemos la matriz de valores de fase, graficarlos en función de la frecuencia.

S. El programa principal

Una vez definidos los pasos a seguir, hay que adentrarse en la programación que debe llevarse a cabo para la implementación del módulo. Existen dos posibilidades, utilizar bloques predefinidos proporcionados por los fabricantes, o implementar manualmente dicha funcionalidad. En nuestro caso, elegimos la segunda opción, ya que de este modo teníamos un mayor control sobre el proceso que se estaba llevando a cabo. Para ello, creamos un Subvi, cuyas entradas provienen de la inicialización del instrumento (osciloscopio), y como salida proporciona el desfase temporal $dT(seg)$:

Fig. 5.1 Bloque de desfase temporal

En el programa principal, introduciremos este bloque, que implementa los seis primeros pasos expuestos anteriormente. Seguidamente, aplicaremos los pasos 7 y 8, que se reducen a introducir dos bloques de multiplicación y a la representación de los dos arrays (fase y frecuencia) en un gráfico.

T. Subvi" dt"

El subvi se divide en tres partes, que se ejecutarán secuencialmente.

La primera, una vez inicializado el instrumento, realiza un autoajuste del osciloscopio junto con el posicionamiento en el origen del eje vertical de la onda y la adquisición de las ondas del canal uno y dos.

La segunda parte de la estructura recibe las dos matrices de onda (entrada y salida) y calcula el punto en que la onda de entrada realiza la transición por cero de positivo a negativo. Ya que la onda de salida no se posiciona en el cero del eje vertical, debemos calcular cual es "su cero", obteniendo el valor máximo, el mínimo y dividiendo su diferencia entre dos.

En la tercera parte, obtendremos el índice del paso por "cero" de la segunda onda. Imponiendo la condición de que sea de positivo a negativo y con un índice superior a la onda de entrada, asegurándonos que se coge siempre el inmediatamente posterior cuyo cambio de signo sea idéntico.

En el procesamiento de las dos ondas, lo que obtenemos es el punto en que se ha producido el paso por cero, conociendo el espacio temporal existente entre cada punto, sabremos mediante una simple multiplicación el desfase expresado en segundos entre las dos ondas para una determinada frecuencia.

XIII. OBTENCIÓN DEL MÓDULO DE LA RESPUESTA FRECUENCIAL

En este apartado se detallarán los pasos seguidos para la obtención y representación del módulo de la respuesta frecuencial correspondiente al filtro pasa-banda que hemos realizado.

En el primer "frame", realizamos una serie de operaciones que pueden parecer simples a primera vista pero que nos resultan imprescindibles a la hora de manipular los diferentes instrumentos, se trata de las inicializaciones de los mismos, las cuales nos pondrán en funcionamiento la comunicación con los instrumentos y nos permitirán configurarlos de acuerdo con las operaciones que deseemos realizar.

Por lo que respecta al segundo "frame", hay que decir que es el lugar en el que se encuentra la operación en si, la cual pasaremos a explicar. Para la obtención de la magnitud será necesaria la definición de una serie de parámetros, la amplitud de la señal de entrada, el número de pasos a muestrear de la señal de salida, el formato de la escala a muestrear (lineal o logarítmica) y las frecuencias de muestreo alta y baja.

Definiremos una estructura "for", que nos realizará las operaciones, un número 'N' de veces, en nuestro el valor del número de pasos a muestrear. El primer paso será calcular la frecuencia de la señal de entrada a muestrear mediante una ecuación cuyos parámetros de entrada serán el número de pasos a muestrear y las frecuencias alta y baja. Cabe decir que hemos incorporado una estructura "case" que nos realizará la ecuación para la obtención de los valores de escala lineales o logarítmicos dependiendo de el valor del switch incorporado en el panel frontal.

Una vez obtenida la frecuencia a muestrear pasaremos a indicársela al generador de funciones para que nos genere la onda correspondiente, la cual haremos pasar a través del filtro para muestrear la salida. Este muestreo de la salida será precisamente el siguiente paso a realizar.

Los dos pasos explicados anteriormente los realizaremos mediante dos bloques "subVi" anteriormente explicados y en el interior de los cuales configuramos los parámetros del multímetro y del generador de señales la cual le introduciremos el valor del la frecuencia a muestrear y la amplitud de la señal mediante patillas de entrada.

Una vez tenemos el valor de la amplitud de la señal muestreada en RMS pasaremos a convertirla para poder representar el bode.

Este proceso lo repetiremos para todos los intervalos de muestreo mediante la estructura "while", la cual nos dará un array de salida con todos los valores,los cuales graficaremos en función de las frecuencias muestreadas para obtener el diagrama de bode de la ganancia.

En el siguiente "frame" procederemos a cerrar la comunicación con el generador y el multímetro mediante los bloques "close".

XIV. ANCHO DE BANDA ('BW'), FRECUENCIA DE RESONANCIA ('F0') Y FACTOR DE CALIDAD ('Q')

En nuestro programa hemos incorporado la obtención de una serie de parámetros tales como la frecuencia de resonancia, el ancho de bando y el factor de calidad.

La frecuencia de resonancia será aquella para la cual el diagrama de bode de la ganancia obtiene el valor máximo. Por ello, para su obtención hemos hecho uso de un bloque estadístico que nos calculará el índice en el cual se encuentra el valor máximo de ganancia a partir de la señal de las ganancias. Una vez tenemos el índice del máximo, pasaremos a buscarlo en las frecuencias de entrada, la seleccionada será la frecuencia de resonancia.

Para la obtención del ancho de banda, hemos aprovechado el mismo bloque que para el cálculo de la frecuencia de resonancia, al cual le hemos incorporado una nueva salida que nos facilitará el máximo valor de resonancia en dB. Una vez obtenido dicho valor, realizaremos una serie de operaciones con él para determinar el ancho de banda.

En primer lugar, le restaremos 3 dB's a dicho valor e iremos muestreando los valores de la ganancia hasta obtener un valor mayor a éste, tras almacenar ese valor pasaremos a buscar un valor que sea menor a la frecuencia de resonancia menos 3 dB's y pasaremos a almacenarlo también. De este modo hemos obtenido los límites superior e inferior del ancho de banda el cual obtendremos realizando la resta de ambos.

Una vez tenemos el ancho de banda y la frecuencia de resonancia podemos pasar a calcular el factor de calidad "Q", que vendrá definido por la división entre la frecuencia de resonancia y el ancho de banda (fo/BW).

Todos los valores calculados en este apartado quedarán representados mediante indicadores en el panel frontal.

XV. REPRESENTACIÓN DE LOS RESULTADOS

En este apartado veremos los resultados obtenidos tras ejecutar el programa con nuestro filtro pasa banda:

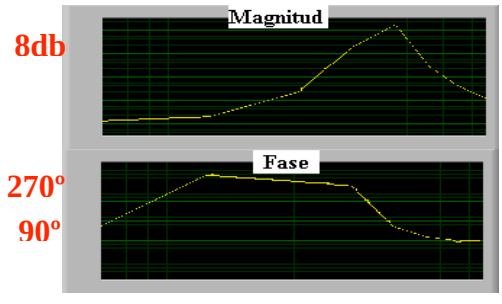


Fig. 8.1 Bloque de desfase temporal

Como podemos ver en la figura 8.1, en la representación de la magnitud, la frecuencia de resonancia se encuentra cercana a los 3308Hz. En este punto la ganancia es cercana a los 8,68db (K=2.71).

La fase, a partir del segundo paso se comporta como cabía esperar, tenemos un filtro con un cero en el orígen, por lo que debe empezar en -90º, o lo que es lo mismo, 270º. Al estabilizarse tras pasar por la década posterior al último polo, bajamos 180º, hasta los 90º.

Por último, mostraremos el fichero (.txt), en donde se guardan los datos de la respuesta frecuencial (magnitud) :

Frecuencia (Hz)	Amplitud (db)
2000,000000	-6,161716
2188,574761	-4,504971
2394,929742	-2,560095
2620,741394	-0,197813
2867,844235	2,787480

XVI. CONCLUSIONES

Hemos podido ver como controlar instrumentos de medida y generación mediante GPIB. La programación visual de Labview facilita su implementación y resulta muy aconsejable segmentar el programa mediante subvi que doten al programa principal de claridad.

En cuanto a los resultados obtenidos, han sido similares a los impuestos analíticamente. Estas diferencias se deben a las no idealidades del circuito unido a la imposibilidad de obtener los componentes con el valor exacto obtenido mediante las expresiones analíticas.

AGRADECIMIENTOS

Esta asignatura ha sido impartida por los profesores Jaume Verd, Bartomeu Alorda y Vicenç Canals (ETG – Departamento de Física de la UIB).

REFERENCIAS

[22] http://www.uib.es/depart/dfs/GTE/education/industrial/ins_electronica_I/index.htm.
[23] LabView 6i : programación gráfica para el control de instrumentación / Antonio Manuel Lázaro.
[24] LabVIEW for data acquisition / Bruce Mihura
[25] Instrumentación electrónica / Miguel A. Pérez García

Eduardo Azcona Soria (Pamplona, 1987) es estudiante de tercer curso de Ingeniería Técnica Industrial (esp. Electrónica Insdustrial) en la UIB.

Jorge Torres Llobera (Palma de Mallorca, 1987) es estudiante de tercer curso de Ingeniería Técnica Industrial (esp. Electrónica Insdustrial) en la UIB.

Pedro Rodríguez Riquero (Palma de Mallorca, 1987) es estudiante de tercer curso de Ingeniería Técnica Industrial (esp. Electrónica Insdustrial) en la UIB.

Disseny Pràctic d'un Radioenllaç Terrenal

Miguel Calvo Fullana

Radiococomunicacions

miguel.calvo.f@gmail.com

Resum— **Es pretén dissenyar un radioenllaç terrenal entre dos punts de la illa de Mallorca, s'analitzaran les condicions inicials de l'enllaç, es seleccionaran els components que formaran el radioenllaç i s'avaluarà la qualitat de manera que compleixi les recomanacions de la ITU-R.**

XVII. INTRODUCCIÓ

Es vol connectar mitjançant radioenllaços un edifici del centre de Palma amb una alçada de 30m situat a la posició: 39º 33' 44" latitud Nord i 2º 39' 11" longitud Est amb l'hotel Amarador de 40m d'alçada situat a 39º 42' 36" latitud Nord i 3º 28' 2" longitud Est. Es tracta d'un enllaç digital de banda ampla (45Mbps) amb una freqüència portadora d'11GHz.

Des de l'hotel Amarador s'utilitza un altre radioenllaç col·locat a la mateixa torre per comunicar-se amb un altre hotel situat a 39º 46' 29,9" latitud N, i 3º 21' 14,5" longitud E. Aquest radioenllaç transmet a la mateixa potència, utilitza les mateixes freqüències que el nostre, i també els mateixos equips però amb una polarització creuada, per tant la seva senyal interfereix amb el nostre radioenllaç.

Podem observar a la Fig. 1 les passes que seguirem per dissenyar el radioenllaç.

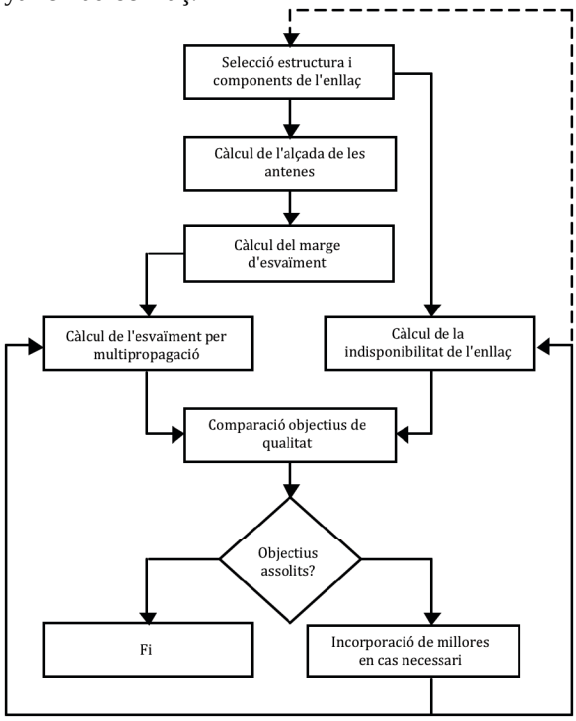


Fig. 3 Diagrama general pel disseny de radioenllaços

Com a suport per a l'estudi i la realització del radioenllaç s'utilitzarà una eina informàtica freeware anomenada Radio Mobile [1].

XVIII. AVALUACIÓ INICIAL

El perfil que uneix Palma amb l'hotel Amarador conté obstacles que bloquegen totalment les zones de Fresnel, com podem observar a la Fig. 2. Per tant, necessitem utilitzar almenys un repetidor.

Fig. 4 Perfil entre Palma i l'hotel Amarador

Analitzem les zones de cobertura visual de les dues estacions i obtenim que no hi ha cap punt de la illa que es vegi des de les dues estacions, per tant necessitem utilitzar un altre repetidor. Podem veure el posicionament final de les estacions a la Fig. 3.

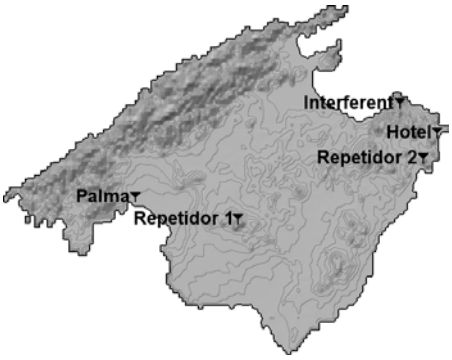


Fig. 5 Posicionament de les estacions

TAULA VI. COORDENADES DE LES ESTACIONS

Estació	Coordenades
Palma	39º 33' 44" N 2º 39' 11" E
Repetidor 1	39º 31' 29,1" N 2º 55' 27,8" E
Repetidor 2	39º 38' 56,6" N 3º 24' 57,2" E
Hotel Amarador	39º 42' 36" N 3º 28' 2" E
Hotel Interferent	39º 46' 29,9" N 3º 21' 14,5" E

S'han posicionat els repetidors de manera que no hi hagi visió directa amb cap estació apart de a la que transmeten, evitant així interferències. A més utilitzarem un pla a quatre freqüències per evitar les interferències en els propis repetidors i d'aquesta manera, ja que no hi ha visió directa que pugui produir interferències (a part de l'altre hotel), utilitzarem una polarització vertical. A més, els repetidors que utilitzarem seran repetidors regeneratius.

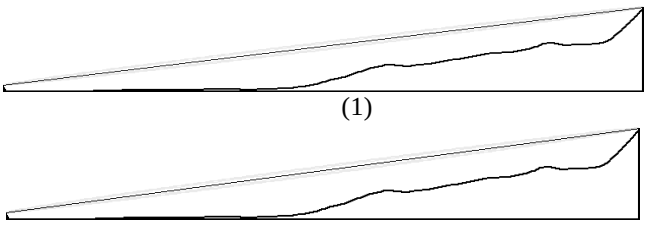
XIX. SELECCIÓ DE COMPONENTS

La guia d'ones que s'utilitzarà és la EW127A [2], del fabricant Andrew. L'antena que s'ha seleccionat per a totes les estacions és la PAR8-107 [3], [4], també del fabricant Andrew, aquesta antena ens proporciona un guany considerable (46,2 dBi) a un preu relativament baix (2.785€). A més, utilitzarem un receptor amb un factor de renou F_r = 10 dB, i una signatura de la que es poden extreure els següents paràmetres del mètode de Mojoli pel càlcul dels esvaïments selectius; per $P_b = 10^{-3}$, B_{eq} = 17 dB, W = 22 MHz i per $P_b = 10^{-6}$, B_{eq} = 13 dB i W = 32 MHz.

XX. ALÇADA DE LES ANTENES

L'alçada màxima de les antenes als edificis de Palma i Cala Ratjada per motius de normativa i estètica és de 5 m.

Si observem els perfils de cada enllaç (Fig. 4-6) no hi ha cap obstacle evident, tot hi així comprovem que l'elevació efectiva de l'obstacle més evident degut a la esfericitat de la Terra no ens provoqui difracció. utilitzant l'expressió (1).



(1)

Fig. 6 Palma - Repetidor 1 (Va 1)

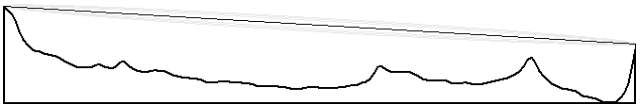


Fig. 7 Repetidor 1 - Repetidor 2 (Va 2)

Fig. 8 Repetidor 2 - Hotel Amarador (Va 3)

TAULA VII. DADES DELS VANS

Paràmetres	Va 1	Va 2	Va 3
Elevació Transmissor (m)	36	445	282
Elevació Receptor (m)	445	282	48
Cota Obstacle (m)	137,9	208	44,1
Dist. Transmissor-Obstacle (Km)	14,1	36,85	4,78
Dist. Obstacle-Receptor (Km)	9,48	7,4	3,29

Aplicant l'expressió (1) i trigonometria bàsica, obtenim que no es requereix elevar les antenes en cap cas. Tot hi així, elevarem les antenes 3m per motius de seguretat.

XXI. MARGE D'ESVAÏMENT

Degut a la freqüència de l'enllaç tenim 0,01 dB/Km de pèrdues per absorció dels gasos atmosfèrics. Com ja havíem observat anteriorment, degut al perfil dels vans no hi ha pèrdues per difracció.

La guia d'ones introdueix pèrdues de 0,1228 dB/m [2]. Les antenes es troben a 3m d'altura, per tant podem aproximar uns 4m de guia d'ones, de manera que tenim 0,5 dB de pèrdues. Els receptors tenen un factor de renou de 10 dB, per tant, el factor de renou del sistema receptor serà de 10,5 dB. A més, hem de tenir en compte que la modulació emprada és QPSK.

U. Marge Brut d'Esvaïment

Per calcular el marge brut, calculem la potència de recepció nominal, i li restem la potència de recepció mínima.

TAULA VIII: CÀLCUL DE MARGE BRUT

Paràmetres		Va 1	Va 2	Va 3
N_0 (dBm)		-163,48		
E_b/N_0 (dB)	$P_b=10^{-3}$	6,78		
	$P_b=10^{-6}$	10,52		
V_b (dB)		76,53		
P_{Rmin} (dBm)	$P_b=10^{-3}$	-80,17		
	$P_b=10^{-6}$	-76,43		
P_{Rnom} (dBm)		-29,56	-35,23	-20,08
MB3 (dB)		50,61	44,94	60,09
MB6 (dB)		46,87	41,2	56,35

V. Marge Net d'Esvaïment

L'estació interferent tan sols té visió directa amb l'hotel Amarador, per tant només afectarà l'enllaç entre el segon repetidor i l'hotel, de manera que per al primer i segon va el marge net és igual al marge brut. El radioenllaç interferent transmet a la mateixa potència, utilitza les mateixes freqüències que el nostre, i també els mateixos equips però amb una polarització creuada

Per calcular la senyal interferent amb l'hotel, la polarització de la nostra antena és vertical, la polarització de la senyal interferent és horitzontal i l'angle d'incidència de la senyal interferent és de 93,3º. Amb aquest angle d'incidència obtenim una discriminació màxima de 65 dB [4] considerant les pèrdues de discriminació de polarització degut als esvaïments, seguint la expressió (2).

$$D_p(F) = D_{p0} - \frac{F}{2} \qquad (2)$$

Calculem el marge net afegint la senyal interferent al renou i tenint en compte que aquest dependrà de l'esvaïment. Obtenim que MN3 = 20,95 dB, i MN6 = 18,46 dB.

XXII. INDISPONIBILITAT DE L'ENLLAÇ

Per calcular la indisponibilitat de l'enllaç, tenim que calcular la indisponibilitat degut a dos factors, la pluja i la falla d'equips.

W. Indisponibilitat Deguda a la Pluja

Observem que la intensitat de pluja excedida durant 0,01% del temps és de 30mm/h a la nostra regió geogràfica [5]. Obtenim els paràmetres necessaris per al càlcul de la indisponibilitat deguda a la pluja [6]. Calculem la atenuació específica de la pluja ($\gamma_{pluja0,01}$), les longituds efectives (le_{pluja}) dels vans i l'atenuació deguda a la pluja ($A_{pluja0,01}$) a cada va. Amb l'atenuació deguda a la pluja, calculem la probabilitat de superar el marge net amb aquesta atenuació.

TAULA IX. CÀLCUL DE LA INDISPONIBILITAT DEGUDA A LA PLUJA

Paràmetres	Va 1	Va 2	Va 3
$\gamma_{pluja0,01}$ (dB/Km)	1,087		
le_{pluja} (Km)	11,465	14,835	5,926
$A_{pluja0,01}$ (dB)	12,46	16,12	6,44
U_{Pi} (%)	0,000010162	0,00019239	0,000077769
U_P (%)	0,000280321		

X. Indisponibilitat per Falla d'Equips

Tenim 6 equips sense protecció, tots tenen el mateix temps mitjà entre falles (MTBF = 60000 h) i el mateix temps mitjà de reparació (MTTR = 7 h). Obtenim una indisponibilitat del 0,0007 %.

Y. Verificació dels Objectius d'Indisponibilitat

Sumem la indisponibilitat deguda a la pluja i la indisponibilitat per falla d'equips i comprovem si compleixen la recomanació [7]. La suma dóna 0,000980321 %, per tant compleix la recomanació ja que és menor que el 0,0336 % màxim recomanat.

XXIII. Fiabilitat de l'Enllaç

Degut a que el nostre enllaç és de gran capacitat (45Mbps), avaluarem l'efecte de l'esvaïment pla i l'esvaïment selectiu.

Z. Esvaïment Pla

Calculem per cada va la probabilitat de que un esvaïment profund superi el marge net (χ_p). Per això hem de tenir en compte que el nostre factor geoclimàtic és K' = $5{,}124 \cdot 10^{-3}$.

TAULA X. Càlcul de l'Esvaïment Pla

Paràmetres		**Va 1**	**Va 2**	**Va 3**
ε_p		17,34	3,68	29
χ_{pi} (%)	$P_b=10^{-3}$	0,00003707	0,002876	0,000743
	$P_b=10^{-6}$	0,0000877	0,006804	0,00131
χ_p (%)	$P_b=10^{-3}$	0,00365607		
	$P_b=10^{-6}$	0,0082017		

AA. Esvaïment Selectiu

Utilitzarem el mètode de Mojoli empíric. Calculem per cada va la probabilitat d'aparició d'un esvaïment multitrajecte (η), el valor mig del retard del raig indirecte (τ_0) i la probabilitat de superar P_b (χ_S)

TAULA XI. Càlcul de l'Esvaïment Selectiu

Paràmetres		**Va 1**	**Va 2**	**Va 3**
P_0		0,36	2,39	0,014
η		0,089	1	0,00917
τ_0 (ns)		0,227	0,583	0,045
T_1 (ns)	$P_b=10^{-3}$	44,6		
	$P_b=10^{-6}$	28,14		
χ_{Si} (%)	$P_b=10^{-3}$	0,0000428	0,00124	0,000000875
	$P_b=10^{-6}$	0,0000988	0,00285	0,00000202
χ_S (%)	$P_b=10^{-3}$	0,001283675		
	$P_b=10^{-6}$	0,00295082		

BB. Verificació dels Objectius de Fiabilitat

Sumem les probabilitats de l'esvaïment pla i l'esvaïment selectiu i comprovem que compleixen la recomanació [8]. Per $P_b=10^{-3}$ obtenim 0,004939745 %, que és menor que 0,006048 %, per tant compleix la recomanació. Per $P_b=10^{-6}$ obtenim 0,01115252 %, que és menor que 0,03584 %, per tant, també compleix la recomanació.

XXIV. Conclusions

Un cop el sistema compleix les recomanacions de indisponibilitat i fiabilitat, podem observar algunes de les decisions preses a l'hora de realitzar aquest sistema.

El punt més conflictiu és l'enllaç entre els repetidors, degut a la seva longitud (44,25 Km) i el fet de que siguin enllaços intermitjos les opcions per dissenyar el sistema es redueixin considerablement. La utilització de repetidors *back-to-back* o no regeneratius queda directament descartada degut a la impossibilitat de complir les recomanacions amb aquests mètodes. També degut a aquest enllaç s'ha de utilitzar un pla a quatre freqüències, ja que amb un pla a dues freqüències i polaritzacions creuades no s'aconsegueix un marge net suficientment elevat en aquest va.

D'altra banda, la utilització de una antena menys potent també dóna problemes, en aquest cas també en el tercer va, on les interferències és fan massa grans com per complir les recomanacions.

Referències

[26] R. Coudé. (2008) Radio Mobile Website [Online]. Available: http://www.cplus.org/rmw/english1.html

[27] "EW127A data sheet," Andrew Corporation, Illinois, USA.

[28] "PAR8-107 data sheet," Andrew Corporation, Illinois, USA.

[29] "PAR8-107 radiation pattern envelope," Andrew Corp., Illinois, USA.

[30] *Characteristics of precipitation for propagation modelling*, ITU-R P.837, 2003.

[31] *Specific attenuation model for rain for use in prediction methods*, ITU-R P.838, 2005.

[32] *Availability Objectives for Real Digital Radio-Relay Links Forming Part of a High-Grade Circuit within an Integrated Services Digital Network*, ITU-R F.695, 1990.

[33] *Error Performance Objectives for Real Digital Radio-Relay Links Forming Part of the High-Grade Portion of International Digital Connections at a Bit Rate Below the Primary Rate within an Integrated Services Digital Network*, ITU-R F.634, 1997.

Assignatura impartida per: Loren Carrasco Martorell

Miquel Calvo Fullana és un alumne d'enginyeria tècnica de telecomunicacions especialitat telemàtica.

Comunicación serie Maestro/Esclavo

Laura Porcel Martín, Juan Carlos Gutiérrez Baños.

Práctica de la asignatura Microordenadores

lauraporma@gmail.com
gutierrezjuanca@hotmail.com

Resumen — **En este trabajo se ha implementado un dispositivo de comunicación serie en configuración MAESTRO/ESCLAVO. Se ha utilizado el microcontrolador PIC16F876 presente en el laboratorio y el programa de control RealPic. El objetivo ha sido el diseño de un protocolo a nivel de capa red sobre un nivel físico basado en el estándar RS232 en su versión asíncrona.**

I. Introducción

Una comunicación Maestro/Esclavo requiere un conjunto de reglas que especifiquen el intercambio de datos u órdenes. Estas reglas definen lo que se conoce como un protocolo de red o también un protocolo de comunicación. En este caso el intercambio se produce entre dos microcontroladores PIC16F876.

Este microcontrolador utiliza a nivel físico el protocolo RS232 encargado de establecer una norma para el intercambio serie de datos binarios entre un equipo terminal de datos y un equipo de Comunicación de datos. La interfaz RS232 está diseñada para distancias cortas, de unos 15 metros o menos, y para velocidades de comunicación bajas, de no más de 20 [Kb/s]. La interfaz puede trabajar en comunicación asíncrona o síncrona y tipos de canal simplex, half duplex o full duplex. En este caso será asíncrona y utilizará un canal simplex en el que el intercambio de datos se realiza en un solo sentido.

A. Objetivos

En este trabajo se han configurado dos entrenadores basados en el PIC16F876 para establecer una comunicación mediante un periférico (USART) que utiliza el protocolo RS232.

Se han implementado tanto una etapa de conexión como una etapa de comunicación de datos entre los entrenadores. La etapa de conexión permite determinar si el entrenador, configurado como esclavo está presente. Este proceso consiste en el intercambio de caracteres específicos. La etapa de comunicación permite al entrenador configurado como maestro enviar los caracteres introducidos en su teclado y al entrenador esclavo recibirlos y mostrarlos en su pantalla LCD. Véase la Figura 1 una fotografía del entrenador con los periféricos utilizados.

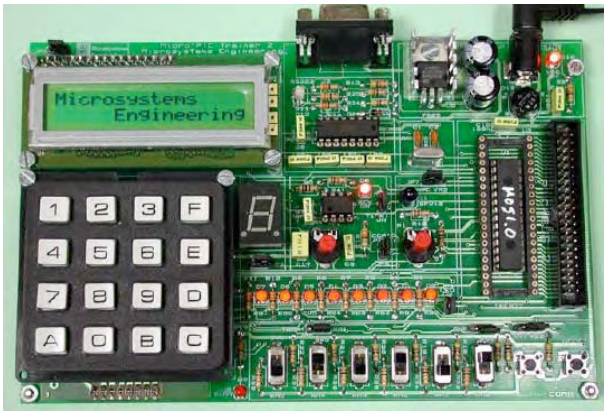


Fig. 1 Fotografía del entrenador utilizado (PIC 16F876).

II. Implementación Hardware

B. Periférico USART.

El periférico USART es el utilizado para la transmisión de datos en formato serie, aplicando técnicas de transmisión sincrónica o asincronica, según su configuración. La característica más destacable de este periférico es que destina un terminal a la transmisión (Tx) y otro a la recepción (Rx), en este caso el sincronismo se hace dentro de cada equipo y la interfaz solo define el uso de un bit de start y otro de stop, para indicar el inicio y fin de transmisión de un byte, es por eso que todos los equipos interconactados deben estar configurados para el mismo bit-rate. Las ventajas más importantes de este modo de comunicación radica en que no se requiere destinar más entradas salidas a completar algunas interfaces como la RS232.

C. Implementación del cable de conexión

Los dos entrenadores se conectan entre ellos cruzando los pines del puerto RS232 de recepción y transmisión y uniendo los pines de tierra. El pin encargado de recibir es el que ocupa la posición 2, el pin de transmisión ocupa la posición 3 y el pin de tierra o gnd es el número 5.

Véase Figura 2 en la que se muestra el esquema básico de la conexión entre dos entrenadores PIC16F876.

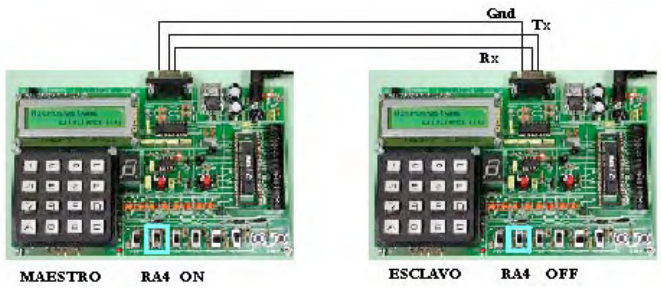


Fig. 2 Esquema de la implementación.

III. IMPLEMENTACIÓN SOFTWARE

El entrenador distinguirá entre los dos modos de operación según la posición del interruptor RA4, nivel bajo para Maestro y nivel alto para Esclavo.

A. Comprobación inicial sobre el interruptor RA4.

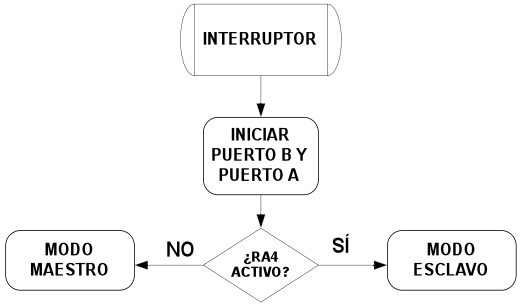


Fig. 3 Diagrama de flujo para seleccionar el modo.

Nuestro sistema de comunicación Maestro/Esclavo se inicia consultando el modo en el que queremos trabajar para una vez decidido comenzar nuestra transmisión de datos. El bucle con que se inicia el programa principal es, en código ensamblador, el siguiente:

```
Loop:       movf    PORTA,W
            andlw   b'00010000'
            btfsc   STATUS,Z
            goto    Vale_0
            movwf   interruptor
            movlw   b'00000000'
            btfsc   STATUS,Z
            goto    Vale_1
Vale_1
            goto InicioEsclavo
Vale_0
            goto InicioMaster
```

B. Modo Maestro.

Inicializaremos este modo con un bucle de petición de conexión, en el que el entrenador enviará cada dos segundos el carácter '@' a través del canal TX a no ser que reciba '=' por su canal RX. Mientras estemos en este bucle el LCD mostrará por pantalla el siguiente mensaje: '…Detectando…'.

Una vez establecida la comunicación, el entrenador enviará por su canal TX el dato introducido en el teclado y mostrará por pantalla el dato enviado y el mensaje '…Conectado…'

Seguimos el siguiente diagrama de flujo:

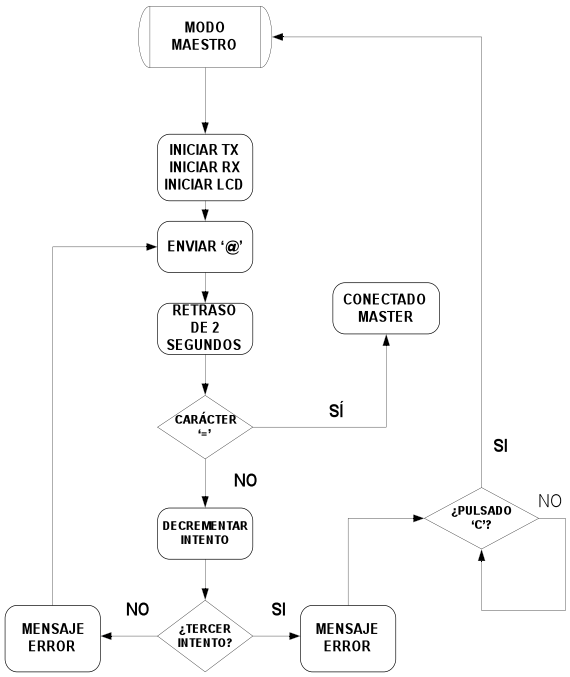


Fig. 4 Diagrama de flujo Modo Maestro.

En código ensamblador quedaría de la siguiente forma:

```
InicioMaster:   call inicio_tx
                call inicio_rx
                call inicio_lcd
                call conectando_lcd
                movlw   0xc5
                call    LCD_REG
                call master_lcd
                movlw  .3
                movwf  intentos

LoopConexionMaster:

                call enviar_@
                call esperar2seg
                call capturar
                call comparar_igual
                call intento
                goto    LoopConexionMaster
```

Utilizando la llamada a subrutinas hemos conseguido construir un programa fácil de entender. A tener en cuenta, destacamos que la función *comparar_igual* nos hará salir del bucle de conexión cuando recibamos el carácter '=', dirigiéndonos al estado conectado el cual veremos a continuación, y la subrutina *intento* al llegar al tercer intento entraremos a un bucle de espera pidiendo reintentar la conexión.

Subrutina c*onectado Maestro e Intento:*

```
ConectadoMaster:
                call    LCD_INI
                movlw   0x80
                call    LCD_REG
                movlw   Mens_1
                 call    Mensaje
                goto enviar
                return
```

```
Intento
        call    error_lcd
        call    esperar1seg
        movlw   0xc5
        call    LCD_REG
        call    master_lcd
        decfsz  intentos
        return
        goto FalloConexion
```

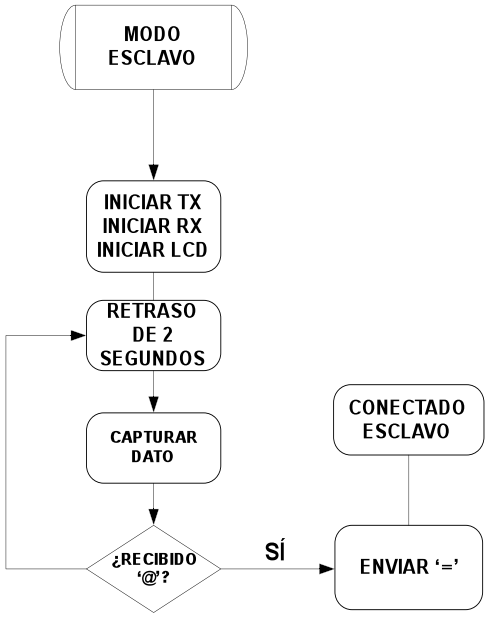


Fig. 5 Diagrama de flujo Modo Esclavo.

C. *Modo Esclavo*

El entrenador, configurado como Esclavo, iniciará un bucle de espera de carácter '@' y mostrará por el LCD el mensaje '…Conectando…'

Una vez conectados ambos entrenadores, nuestro entrenador mostrará por pantalla el valor que reciba por el canal RX además del mensaje '…Conectado…'

Su correspondiente diagrama de flujo se puede observar en la Fig. 5.

Y su código en ensamblador, también con un programa inicial fácil de entender, es:

```
InicioEsclavo:
        call    inicio_tx
        call    inicio_rx
        call    inicio_lcd
        call    esperar_lcd
        movlw   0xc5
        call    LCD_REG
        call    esclavo_lcd
LoopConexionEsclavo:
        call    esperar2seg
        call    capturar
        call    comparar_@
        goto    LoopConexionEsclavo
```

La subrutina *capturar* consultamos el dato recibido en el registro RCREG y lo introducimos en la variable global RECIBIDO.

B. *Parte opcional realizada*

Como elemento opcional añadimos a nuestro sistema de comunicación control de errores en la conexión, limitando este a tres intentos, tras el cual si no conseguimos conectarnos pediremos pulsar la tecla 'C' para reintentar conectarnos.

IV. Conclusiones

En este artículo hemos desarrollado un software capaz de interconectar dos PIC16F87, teniendo en cuenta una fase preliminar de petición de conexión, no es un sistema muy complejo de realizar y sus utilidades en este caso también son escasas, aunque si cabe destacar que es la base principal para poder implementar un software basado en controladores PIC con una capacidad de comunicación muy superior.

Referencias

[34] Microchip PIC16F87X Data Sheet, 28/40-Pin 8-Bit CMOS FLASH Microcontrollers.
[35] Tomeu Alorda, Apuntes de la asignatura de Microcontroladores, de 2º Telemática, UIB.
[36] Mikel Etxebarria, (c) Microsystems Engineering (Bilbao), ejemplos Real Pic.
[37] William Stalings, "Comunicaciones y redes de computadores", 6ª ed., Pearson Education, 2000.

Asignatura impartida por Bartomeu Alorda y Pere Pons

Juan Carlos Gutiérrez Baños
Bachillerato en Colegio San Pedro.
Estudiante de 3º de Ingeniería Técnica de Telecomunicaciones esp. Telemática.

Laura Porcel Martín
Bachillerato en I.E.S. Guillem Sagrera.
Estudiante de 3º de Ingeniería Técnica de Telecomunicaciones esp. Telemática.

DISSENY DEL JOC DE DAUS "CRAPS"

Christian Peter Winter, Francisco Muñoz Contreras

Segon curs d'Enginyeria Tècnica Industrial, Especialitat en Electrònica Industrial
christian@kpwinter.com
feco_m@msn.com

Resum— **"Craps" està present a tots els grans casinos del món i es un dels jocs de daus amb més èxit. Llavors, la proposta es dissenyar e implementar sobre una placa programable una versió digital d'aquest joc. Les eines necessàries per dur a terme aquesta tasca seran el programa de disseny MAX+plus II i la placa d'entrenament UP-1, tots dos productes de l'empresa Altera (una de les empreses líder en tecnologies de lògica programable).**

V. INTRODUCCIÓ AL JOC

Inicialment, el jugador fa l'aposta que cregui oportuna i llança els dos daus. L'objectiu es treure a la primera tirada (també anomenada tirada de sortida) un resultat total igual a 7 ó a 11 de manera que el jugador guanya dues vegades l'aposta feta. Es perd si es treu un "crap", es a dir, un 2, 3 ó 12.

Si el tirador obté un 4, 5, 6, 8, 9 ó 10 es diu que ha tret el seu "punt". A partir d'aquest moment, l'objectiu serà tornar a treure el mateix "punt" per recuperar l'aposta (per exemple si s'ha tret un 4, s'ha d'intentar tornar a treure un 4). Si es treu un 7 abans de treure el "punt", es perd (també es diu "crap").

Aquestes són les regles més bàsiques del joc. Per obtenir una informació més completa:

Link: http://es.wikipedia.org/wiki/Craps

VI. INTRODUCCIÓ A LA PLACA

La versió electrònica del joc consistirà en el maneig d'una placa programable (ex. Fig.1). El sistema constarà de tres polsadors: Dos per emular el llançament dels daus y un altre (de reset) per inicialitzar la partida. A més, els polsadors dels daus es podran pitjar de manera independent.

Fig. 1 Exemple d'una placa d'entrenament UP-1.

Els resultats es mostraran a dos displays de set segments (ex. Fig.2). La informació visualitzada en aquests displays seguirà una pauta predefinida de la següent forma:

Primerament es mostrarà "In" fins que es polsin els polsadors dels daus, moment en el qual s'observarà als dos displays una fluctuació aleatòria de valors entre 1 i 6 (les cares d'un dau) que al final s'estabilitza a un valor concret per a cada dau. Finalment, passats cinc segons mostrant els valors, es visualitzarà "Gn" si el jugador ha guanyat, "Cp" si ha perdut o "Pn" si ha tret un punt (en aquest cas s'haurà de tornar a polsar). Per reinicialitzar, bastarà amb polsar el botó de reset i el sistema tornarà al seu estat inicial.

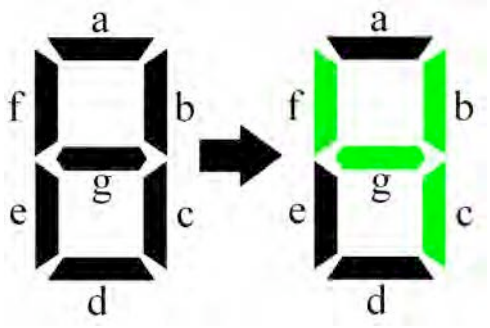


Fig.2 Exemple d'un display de set segments. S'observa que el display té distints segments (a, b, c, d, e, f í g) que, controlats individualment, donen lloc a multitud de combinacions (entre altres els números del 0 al 9).

VII. FUNCIONAMENT INTERN DEL DISSENY

C. Màquina d'estats

Tot disseny seqüencial es pot definir a partir d'una màquina d'estats. Aquesta màquina té per funció representar el funcionament intern del sistema a nivell d'estats y transicions assignant a cada estat les sortides corresponents y a cada transició les condicions específiques.

Exemple senzill: Tenim un sistema format per una bombeta y un interruptor. Per tant, tenim 2 estats: "bombeta apagada" y "bombeta encesa". Si estam a l'estat "bombeta apagada", la sortida "donar tensió a la bombeta" està a 0 (no donam tensió). Si ara polsem el interruptor (que és la condició), passarem al estat "bombeta encesa" estant la sortida a 1 (donam tensió).

La vertadera utilitat d'una màquina d'estats s'observa en sistemes més complexos en els quals ens topam amb desenes d'estats, sortides i condicions.

En el nostre cas, la màquina (Fig.3) ja es relativament complexa i encara que pot resultar difícil entendre-la completament, segueix essent la millor manera per representar de forma visual com funciona el disseny.

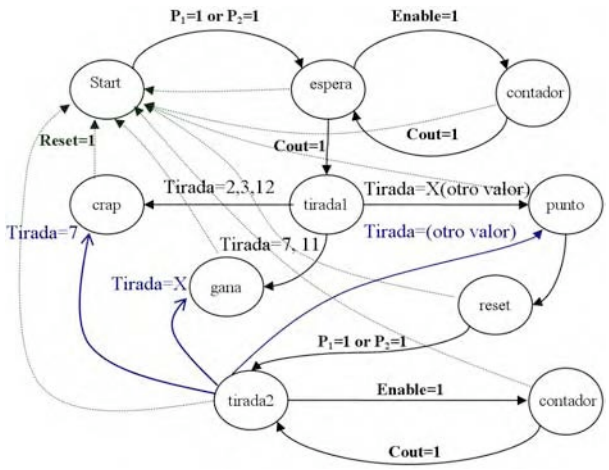


Fig. 3 Màquina d'estats que controla el funcionament del sistema. Podem observar que està formada per 10 estats (que són els 10 blocs que integren el disseny) i que les transicions depenen de diferents condicions a l'hora.

D. Disseny amb el programa

Una vegada ideada la màquina d'estats que defineix el funcionament del sistema, passam al disseny mitjançant un programa especial. Aquest programa (anomenat MAXplus II) es una eina molt potent i té, entre altres moltes funcions, un editor de codi (Fig.5), un editor gràfic (Fig.4), un compilador i un simulador. Per tant, es possible editar diferents blocs lògics a nivell de codi y llavors interconnectar-los amb l'editor gràfic. Finalment es possible compilar y simular el sistema conjunt y així verificar el funcionament correcte del disseny.

Els distints blocs creats (amb l'editor de codi) per implementar el joc de "Craps" son els següents:

- Comptadors de 1 a 6 (Fig.4, Fig.5) .
- Temporitzadors.
- Habilitadors.
- Multiplexors.
- Divisors de freqüència.
- Sumador.
- Controlador.
- Conversors BCD integrats.
- Pautes pseudo-aleatòries implementades.
- Màquina d'estats principal implementada.

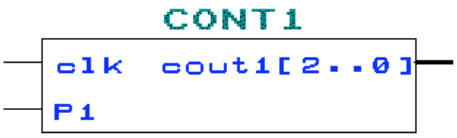


Fig. 4 Comptador editat textualment (Fig.5) e implementat a l'editor gràfic en forma de bloc.

E. Explicació breu de la funció de cada bloc

Per entendre correctament el funcionament del sistema complet, és necessari entendre la funció que té cada bloc de l'integrat.

Com que hi ha un total de 10 blocs diferents, una explicació exhaustiva de cada bloc seria excessiu. Per aquest motiu, s'ha optat per una explicació qualitativa.

1_Contador de 1 a 6:

En el període de temps que el polsador es troba polsat, el comptador està habilitat i compta a una freqüència de 25,175MHz (aquest es el seu rellotge intern). Tenint en compte que la freqüència es molt elevada, podem suposar que el valor que finalment es copia a la sortida (quan deixam de polsar) es completament aleatori.

2_Temporitzador:

La funció del temporitzador es comptar el temps que s'ha polsat el polsador i, quan s'ha amollat aquest, posar a la sortida un valor proporcional al temps comptat (temps multiplicat per 5). El valor màxim que es posarà a la sortida serà de 10 segons (si s'ha polsat el polsador 2 o més segons).

3_Habilitador:

Té per missió habilitar altres blocs depenent de si una variable interna es igual o no a una de les seves entrades. Principalment serveix per organitzar la tasca a executar.

4_Multiplexor:

Segons el valor que tingui a la entrada selectora, selecciona una de les seves entrades i la copia a la sortida. D'aquesta manera es selecciona la sortida del comptador o la pauta pseudo-aleatòria segons convingui.

```
--contador 6
ENTITY cont1 IS
    PORT(clk,P1: IN BIT;
         cout1:OUT INTEGER RANGE 1 TO 6);
END cont1;

ARCHITECTURE arch1 OF cont1 IS
BEGIN
    PROCESS (clk)
    VARIABLE n: INTEGER RANGE 1 TO 6;
        BEGIN
            IF (clk'EVENT AND clk='1') THEN
                IF P1='1' AND n<6 THEN
                    n:=n+1;
                ELSIF P1='1' AND n=6 THEN
                    n:=1;
                ELSE
                    cout1<= n;
                END IF;
            END IF;
        END PROCESS;
END arch1;
```

Inicialitzacions

Comptador 1-6 dependent de E1

Fig.5 Exemple de Codi. Text escrit amb l'editor de codi que implementa un comptador de 1 a 6 que s'habilita amb una entrada E1 i s'incrementa amb cada període de rellotge.

5_Divisor de freqüència:

Com ja ho diu el nom, divideix per 10 la freqüència que li entra. Com que la freqüència d'entrada es molt elevada (25,175MHz), resulta necessari dividir-la per poder dur a terme certes tasques.

6_Sumador:

Suma les seves entrades. Serveix per sumar els valors obtinguts en el llançament dels daus (emulat per el polsadors).

7_Controlador:

Necessari per saber si ja s'han polsat els polsadors. Segons la combinació (ningú polsat, un polsat, tots dos polsats) activa un senyal o un altre.

8_Conversor BCD integrat:

S'ocupa de controlar els displays de set segments de manera que es visualitza la informació correcta.

9_Pauta pseudo-aleatòria:

Abans d'estabilitzar-se el valor, aquest estarà fluctuant de manera aleatòria entre valors de 1 i 6. Realment, aquesta fluctuació no es aleatòria (està controlada per una màquina d'estats interna).

10_Màquina d'estats principal:

Controla tota la tasca de inici a fi. Segons les senyals que activa, el sistema fa una cosa o l'altra (entre altres, decideix si s'ha guanyat, perdut o tret un punt).

El procés que duu a terme el circuit es, en general, seqüencial però també hi ha tasques que es duen a terme de manera paral·lela. A nivell d'exemple:

L'entrada P1 (polsador 1) habilita el temporitzador, aquest l'habilitador que, en conseqüència, habilita la pauta aleatòria. Aquesta passa per el multiplexor y quan acaba, habilita el sumador y la màquina d'estats principal. Finalment, quan aquests han processat la informació, habiliten el conversor BCD obtenint així el resultat final. Es veu que és un procés totalment seqüencial. De manera paral·lela, es duu a terme el mateix procés per a l'entrada P2.

Aquest exemple fa clara la complexitat del disseny. No és d'estranyar que un sistema tan complex presenti certs errors. Per aquest motiu, el programa inclou dues funcions merament importants: el compilador i el simulador. Aquestes funcions comproven tot el codi i, en cas d'errors, informen al programador. Si no hi ha cap tipus d'error, ja es pot passar a l'últim pas que és la implementació sobre una placa.

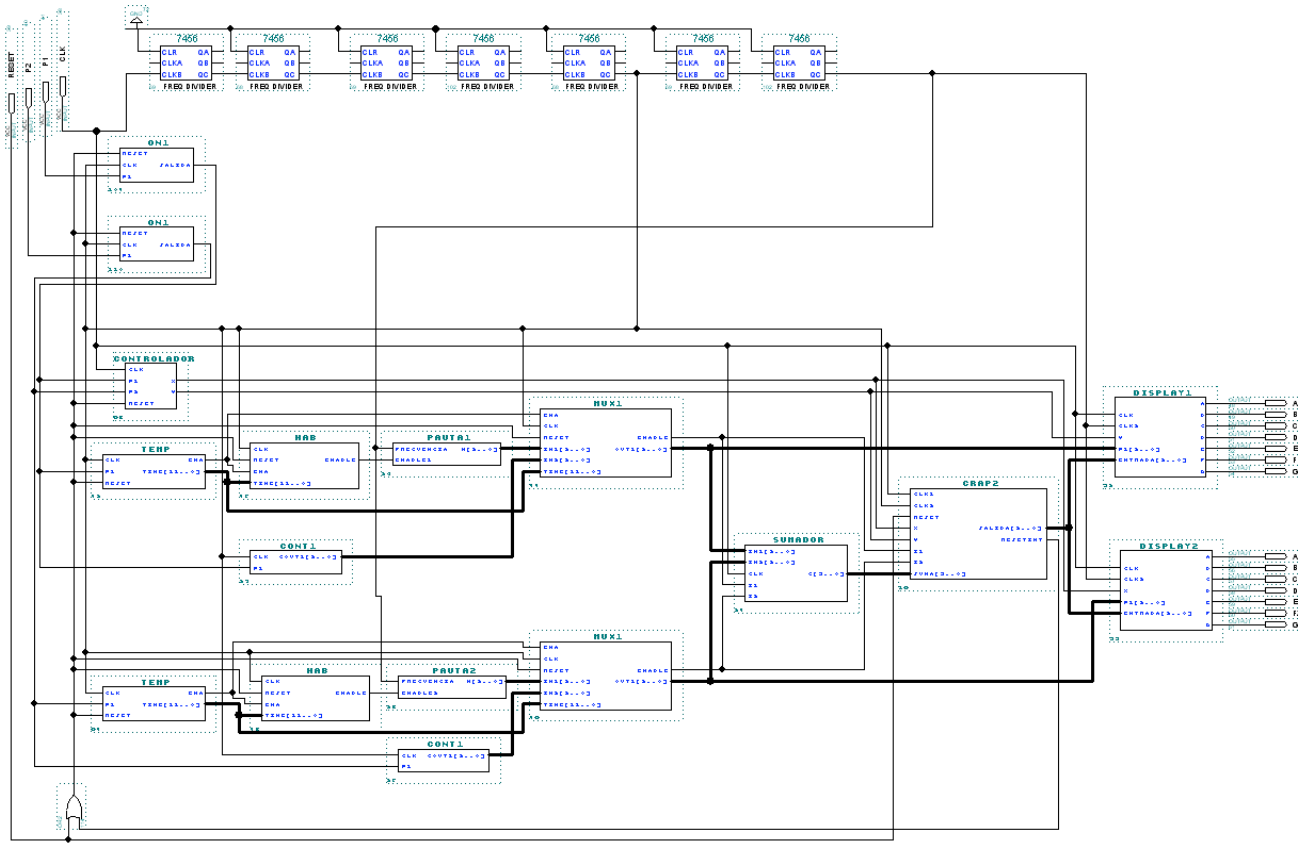


Fig. 6 Esquema del sistema complet (dibuixat amb l'editor gràfic)

F. Disseny final

Tenint tots els blocs dissenyats, es pot passar a la interconnexió d'aquests. Mitjançant l'editor gràfic podem "dibuixar" els blocs i , llavors, connectar-los entre ells amb diferents tipus de línies (d'un bit o de múltiples bits). Fet això, el disseny estarà complet (Fig. 6) y llest per ser simulat i implementat en la placa.

VIII. IMPLEMENTACIÓ FÍSICA

La implementació física es l'últim pas del disseny. En primer lloc es connectarà una "byte-blaster" (Fig. 7) com a medi de comunicació entre PC i placa. La resta és una configuració que es duu a terme pas a pas y ve explicada en el datashett (full de característiques) de la placa en qüestió.

Resumint, els passos a seguir son aquests:

- Assignar la placa →→→ EPF10K20RC240-4

- Compilar
- Assignació de pins
- Compilar (creació d'un fitxer .sof)
- Programar placa (amb la funció "programmer")

Seguits aquest passos, la placa ja es trobarà programada i llesta per a ser utilitzada, facin les seves apostes!

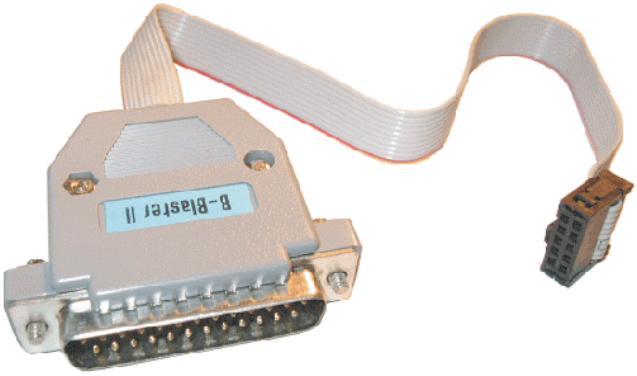


Fig. 7 Fotografia d'una "Byte-blaster". Aquest cable es connectarà entre el port paral·lel del PC i la placa programable (es la connexió física entre els dos components).

IX. CONCLUSIONS

Encara que el joc de Craps real pugui ser bastant més entretingut que no la seva versió digitalitzada, amb aquest disseny queda clar que, encara que una tasca sigui complexa, amb les eines adequades es possible implementar-la sense grans dificultats.

AGRAÏMENTS

Volem donar les gràcies a la Universitat de les Illes Balears per proporcionar-nos les eines de disseny, i la placa d'entrenament i al professor de l'assignatura "Sistemes electrònics digitals", Josep Lluís Rosselló Sanz que ens va ajudar en moments de dubte.

Mesura de Distàncies amb Ultrasons

Cristian Peter Winter, Francisco Muñoz Contreras[2]

Pràctica corresponent a l'assignatura d'Instrumentació Electrònica I

Resum— **L'ús de la tecnologia basada en ultrasons és cada vegada més present a les nostres vides. Un exemple és el seu ús als aparcaments amb ajuda d'una il•luminació per tal de saber si hi ha una plaça lliure, també als paraxocs dels cotxes per saber la proximitat del cotxe aparcat que hi ha al darrera, i moltes més. Per aquest motiu comprovarem com és el funcionament d'aquest sistema i dels sensors ultrasònics al laboratori..**

X. Introducció

Els sensors utilitzats consten d'un emissor i d'un receptor d'ultrasons, el primer envia un senyal específic el qual rebotarà amb l'objecte que volem mesurar la seva distància, i una vegada rebotat, rebrà el senyal (també anomenat eco) el receptor. El temps transcorregut des de que s'emet el senyal fins que arriba al receptor és la variable clau per tal de calcular la distància de l'objecte(Figura1), ja que es segueix la següent fórmula

$$d = \frac{ct_1}{2} \qquad (1)$$

on c és la velocitat del so, que en l'aire a 25ºC és $c \approx 343 m/s$. La figura 1 mostra aquest procés de mesura de distàncies.

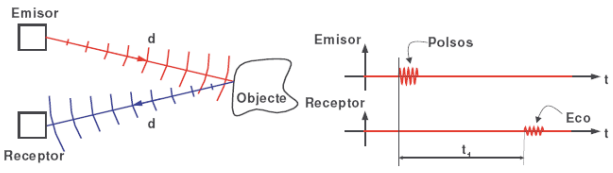


Figura 1. Detecció distància amb ultrasons.

XI. Material utilitzat

Els elements utilitzats en aquesta pràctica són els que s'enumeren a continuació:

- Transductors d'ultrasons: emissor 400ST160 i receptor 400SR160 del fabricant Prowave.
- Oscil·lador astable basat en un integrat NE555.
- 2 Amplificadors operacionals TL071.
- Díode 1N4148.
- Transistor bipolar BJT Q2N2222A.
- Integrat 74HC132 de portes lògiques NAND.
- 5 condensadors (3 de10nF, 1µF i 33nF).
- 2 potenciòmetres (1kΩ i 5kΩ).
- 8 resistències (2 de10kΩ, 2 de 1kΩ, 47kΩ, 1MΩ, 330Ω i 120Ω).
- Protoboard (també anomenat placa de proves).

Equips d'Instrumentació i mesura utilitzats al laboratori:

- Font d'alimentació de continua (+5V i ± 15 V).
- Generador de funcions d'alterna.
- Oscil·loscopi i sondes.
- Multímetre digital.

Si no sabeu per a que serveixen o com funcionen alguns dels elements descrits anteriorment trobareu informació a internet o en qualsevol altra font, com per exemple a " http://es.wikipedia.org ".

XII. Esquema de funcionament

Per tal de comprendre com funciona el circuit complet s'analitzaran primer per separat els quatre blocs més importants del circuit.

A. Habilitador dels polsos.

S'encarrega de controlar el generador de polsos de senyal per a que aquests s'emetin durant 750µs cada període de 0.7s (aproximadament) utilitzant el NE555 com a oscil·lador astable.(Figura 2 a). En definitiva aquest circuit s'encarrega d'implementar una finestra temporal d'activació del sensor.

B. Generador de polsos.

S'encarrega de crear polsos de 40kHz que excitaran l'emissor d'ultrasons amb ajuda del transistor BJT per amplificar el senyal. (Figura 2 a)

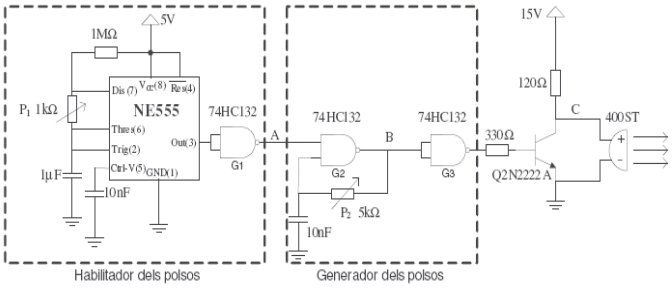


Figura 2a. Blocs d'excitació de l'emissor.

C. Filtre i amplificador.

La seva funció és la de rebre el senyal proporcionat pel receptor, amplificar aquest senyal i filtrar amb precisió només les components que superin els 15kHz-20kHz i així no deixar passar soroll no desitjat que distorsioni la senyal enviada per l'emissor.(Figura 2b)

D. Detector d'envolvent.

Detecta l'envolvent de l'eco de la fase anterior i dóna la senyal de sortida Vo.(Figura 2b)

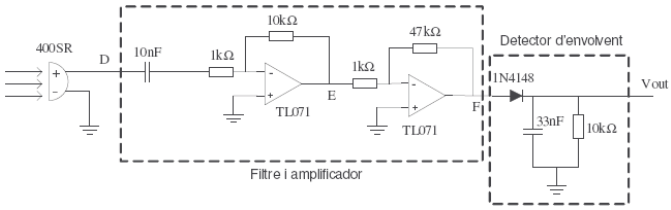


Figura 2b. Blocs de recepció de senyal.

XIII. Muntatge

En aquest apartat s'explicaran les passes a seguir per tal d'anar muntant el circuit i comprovar en cada bloc si aquest

funciona, ja que si ho muntem tot a la vegada en el cas de que hi hagi qualque component romput o mal connectat, seria més difícil de trobar l'error. També es proporcionaran els resultats que hem obtingut nosaltres al laboratori. Per saber quin és el patillatge del NE555, el BJT i dels amplificadors operacionals hem d'obtenir els seus "datasheets" (o full de característiques) a la pàgina web del fabricant o pàgines específiques en "datasheets".

A. Habilitador de polsos.

S'ha de muntar a la protoboard el NE555 tal i com es mostra a la figura 2a. Després visualitzant amb la sonda i l'oscil·loscopi el node A, anirem variant el valor del potenciòmetre per tal d'aconseguir a aquest punt un valor de nivell alt (és a dir a tensió alta) durant 500µs.

Els valors que hem obtingut han estat, un pols de 500µs i valor V=5V (nivell de referència connectat al NE555= nivell alt). També ens hem fitxat com es carrega i descarrega el condensador d'1µF amb un període de 820ms.

B. Generador de polsos.

Seguint les especificacions de la figura 2a, es munta el circuit, però enlloc de connectar en el node A el bloc "habilitador de polsos" el connectarem directament a 5V. Ara visualitzant amb la sonda el node B, ajustarem el potenciòmetre per tal de tenir una senyal de 40kHz, i després farem el mateix pel node C.

Una vegada fet això podrem interconnectar els dos blocs i l'emissor d'ultrasons. Si ara tornam a mesurar el node C comprovarem que cada 0.7s aproximadament es generen 20(± 1) polsos de 40kHz.

Les nostre mesures obtingudes a aquest apartat han estat al senyal B un Voltatge=5.2V, freq=40kHz, Vrms=3V. Després pel que fa al node C, obtinguérem un V=15V (tensió de referència) i freq=40kHz, que representaven 20 polsos de senyal. El que es mostra a la figura 3 a l'oscil·loscopi és el que es representa al node C.

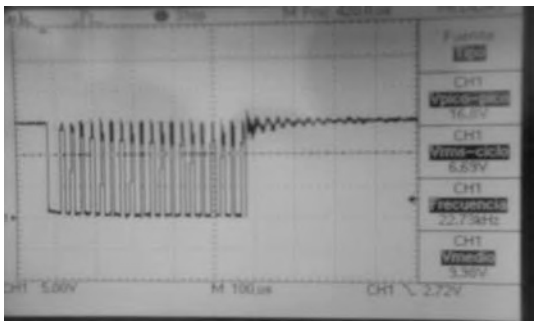


Figura 3. Senyal al node C obtingut amb l'oscil·loscopi.

C. Filtre i amplificador de l'eco.

El següent pas és muntar aquest mòdul però sense connectar encara el receptor d'ultrasons, enlloc d'això, connectarem el generador de funcions amb una ona sinusoïdal de 1V de pic a pic, un offset(o sobrepassament) de 250mV i una freqüència 40kHz. Després s'ha de pitjar l'atenuador de 20dBs al generador de funcions i mesurar el node D i E. Una vegada obtinguts els valors s'ha de fer un escombrat de freqüències per tal de determinar la seva freqüència de tall (f_{-3dB}), aquesta és la freqüència on el guany o amplificació del filtre a caigut -3dB, o el que és el mateix s'ha reduït en un factor $1/\sqrt{2}$, això determina a partir de quines freqüències el filtre deixa passar el senyal i quines altres no en deixa per tal d'evitar transmetre sorolls de senyal no desitjats.

Finalment, una vegada preses les mesures, aplicarem l'atenuador de 40dBs i mesurarem la sortida de l'amplificador al node F, per després poder calcular el guany K de l'amplificador que correspon a l'expressió

$$\frac{Vo}{Vi} = K \qquad (2)$$

En aquest apartat vàrem generar un senyal al node D amb un voltatge pic a pic de 950mV, freq=40kHz i valor mitjà de 237mV. Després de anar variant les freqüències trobarem un valor màxim de sortida a freqüència de 51kHz de V_{pp}=8.9V. Per tant si volem saber quan ha caigut -3dBs hem de fer $\frac{8.9}{\sqrt{2}} = 6.29V$, ara fitxant-nos a l'oscil·loscopi i variant la freqüència trobàrem que el valor V_{pp} =6.29V correspon a una freqüència de 13.4kHz.

D. Detector d'envolvent

En aquest darrer bloc es connectarà el node F al detector d'envolvent, i es comprovarà com varia la tensió a Vout quan variam l'entrada del generador de funcions al node D.

Si totes les passes anteriors han estat correctes ara podem llevar el generador de funcions i connectar el receptor d'ultrasons. L'emissor i el receptor han d'estar orientats en la mateixa direcció. Finalment es posarà un objecte en front seu (exemple una carpeta) a una distància d'uns 40cm. El nostre muntatge a la protoboard va ser finalment com es mostra a la figura 4.

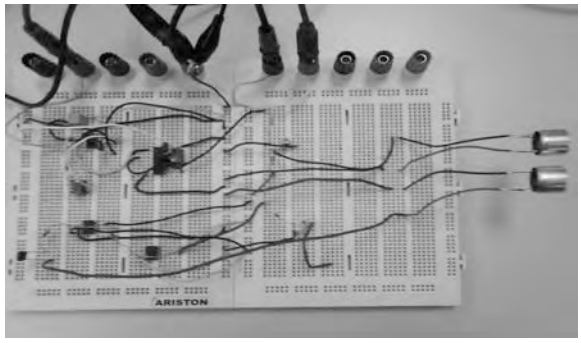


Figura 4. Muntatge a la protoboard.

Finalment s'observa amb l'oscil·loscopi i dues sondes els polsos generats per l'emissor d'ultrasons (node C) i l'envolvent de polsos (Vout). Per això s'ha d'ajustar l'escala temporal de l'oscil·loscopi a 500µs/divisió . Amb la utilització de la funció "cursor" de l'oscil·loscopi per mesurar el temps des del començament de l'emissió dels polsos i l'inici de l'envolvent. Una vegada anotat aquest temps ho introduïm a l'equació 1 (anotada al principi del document) i trobarem així la distància del nostre objecte.

S'ha de tenir en compta, que a vegades al rebre l'envolvent apareixen dues ones, la primera i més petita sol ser per l'acció del rebot del senyal a la taula o altre objecte, el que s'ha de tenir en compta a l'hora de mesurar és la segona ona que té una amplitud major.

Nosaltres vàrem posar l'objecte a 35cm de distància. El temps transcorregut entre les dues senyals mesurades amb els

cursors va ser de 2.28ms, introduint aquesta dada a l'equació trobam que la distància és de 38cm, que s'aproxima molt al valor real.(Figura 5)

XIV. QÜESTIONS TEÒRIQUES

En aquest apartat analitzarem quatre qüestions teòriques per tal de tenir una visió sobre l'aproximació dels càlculs previs, realitzats analíticament, i els resultats obtinguts al laboratori.

1. Calcular el cicle de treball δ del senyal periòdic de sortida del NE555.

Com el càlcul de les expressions seria massa complex i llarg, se donarà les expressions finals obtingudes per tal de calcular el cicle de treball. El càlcul en qüestió s'ha realitzat a les classes de problemes.

temps a nivell alt: $t_H = 0.693 \cdot (R1 + R2) \cdot C1$ (3)

temps a nivell baix: $t_L = 0.693 \cdot R2 \cdot C1$ (4)

relació nivell alt-baix: $\delta = \frac{t_H}{t_H + t_L} = \frac{R1 + R2}{R1 + 2 \cdot R2}$ (5)

Sabent aquestes expressions podem ara respondre a la qüestió. Directament substituint en (5) obtenim δ=0.99.

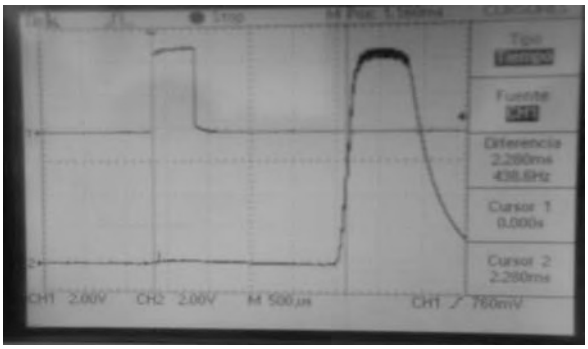


Figura 5. Ona enviada per l'emissor i ona rebuda pel receptor a Vout.

2. Comentar si es podria eliminar G1 (connectar directament la sortida del NE555 a G2) si feim que el NE555 generi un senyal amb cicle de treball 1-δ, en comptes de δ.

Pensant que la sortida del NE555 ens proporciona un senyal, el qual està més temps a nivell alt que baix, podríem suposar en canviar el δ calculant abans per 1-δ, i així estalviar-nos una porta lògica NAND, si això fos possible la nostra suposició seria correcta, però si ens fitxam bé en l'equació (5), si intentam posar una de les resistències molt petita i altra molt gran arribarem a la conclusió de que amb aquesta configuració del NE555 el nostre DutyCycle (δ) només pot variar entre $0.5 \leq \delta \leq 1$ i per tant la porta NAND és necessària.

3. El circuit de recepció presentat utilitza un filtre passa-altes de guany K seguit d'un amplificador de guany A. Comentar si seria viable posar un únic amplificador de guany K·A.

Per poder respondre a aquesta pregunta hem de conèixer primer el significat d'ample de banda i de guany a un amplificador.

El guany d'un amplificador ho hem descrit anteriorment i correspon a l'equació (2), que representa el factor multiplicatiu que s'amplifica el senyal d'entrada.

L'ample de banda és el rang de freqüències en el qual l'amplificador ens dóna un guany constant.

Una de les característiques que tenen els amplificadors és que quan volem augmentar el guany doncs disminuïm l'ample de banda i viceversa. El que ens proporciona posar dos amplificadors en cascada (en sèrie) és que si volem augmentar el guany, l'ample de banda no es veu tan perjudicat com tenint un tot sol, per aquest motiu és necessari tenir dos amplificadors operacionals enlloc d'un.

4. Observar l'efecte que té dins el circuit el disminuir o augmentar el valor del condensador del detector envolvent.

El que ens proporciona augmentar el condensador de sortida és que petits canvis de tensió (arrissat) fa que la sortida es mantingui constant, i si per exemple passem d'un valor alt a un baix de tensió en un instant de temps molt petit, enlloc de tenir un canvi brusc tindrem que la tensió va davallant poc a poc.

XV. CONCLUSIONS

La realització d'aquesta pràctica ens ha ajudat a comprendre el funcionament d'aquests tipus de dispositius i que encara que un sistema de detecció com aquest pareixi una tasca molt complicada, podem veure que amb un poc d'informació i ganes d'aprendre se pot realitzar circuits molt útils per a diverses aplicacions.

Assignatura impartida per: Miquel Roca i Vicenç Canals

Christian Peter Winter és alumne de la titulació d'Enginyeria Tècnica Industrial (especialitat en electrònica industrial) i ha cursat segon curs de la titulació durant el curs 2007-2008. El treball aquí presentat es correspon amb l'informe d'una pràctica de l'assignatura d'Instrumentació Electrònica I, impartida pels professors Eugeni Isern, Miquel Roca i Vicens Canals.

Francisco Muñoz Contreras és alumne de la titulació d'Enginyeria Tècnica Industrial (especialitat en electrònica industrial) i ha cursat segon curs de la titulació durant el curs 2007-2008. El treball aquí presentat es correspon amb l'informe d'una pràctica de l'assignatura d'Instrumentació Electrònica I, impartida pels professors Eugeni Isern, Miquel Roca i Vicens Canals.

Relació de PFC presentats el curs 2007/2008

ARQUITECTURA TÈCNICA

- Esteban Font Amengual, "Guia de consulta para el arquitectotécnico al servicio del promotor", dirigit per Mateu Moyà Borrás (setembre 2007)
- Juan Femenia Price, "La casa bioclimática", dirigit per Salvador Juan Mas (març 2008)
- Ignasi Fuster Marí, "Estudi i proposta d'intervenció d'una estructura sitiuada al Port d'Alcúdia exposada a l'ambient marí", dirigit per Joan Muñoz Gomila (desembre 2007)
- Matias Lladó Matas, "Intervenció a forjats unidireccionals de fusta", dirigit per Joan Muñoz Gomila i Gabriel Horrach Sastre (juliol 2008)
- Ana Oliver Rodríguez, "Gestió i control dels residus de construcció i demolició a obres d'edificació", dirigit per Victoriano García Martínez (octubre 2007)
- Emilio José Muñoz Riscos, "Guía para la implementación de un sistema de gestión de la calidad basado en la norma UNE-EN ISO 9001: 2000 para pequeña empresa constructora", dirigit per Victoriano García Martínez (octubre 2007)
- Laura Summers Ribot i Ana Derqui Carrillo, "Procedimiento para seguimiento de control de ejecución basado en el código técnico de la edificación", dirigit per Joan Muñoz Gomila i Mateo Moyá Borrás (febrer 2008)
- María Inés Burguera Burguera i Neus Monserrat Bauza Obrador, "Molins de vent de Mallorca", dirigit per Salvador Juan Mas (abril 2008)
- Guillermo Bauza Alomar, "Manual de ayuda para instalaciones y presupuestos de los capítulos de electricidad, gas y telecomunicaciones", dirigit per Gabriel Horrach Sastre (abril 2008)
- Lluisa Escandell Pastor i Maria Magdalena Melis Sancho, "Estudi de dosificacions de formigons confeccionants a obra", dirigit per Gabriel Horrach Sastre (abril 2008)
- Francisco Jaime Nieto Vallespir, "Revisión, actualización y rediseño de las fichas de uso y mantenimiento COAATM", dirigit per Gabriel Horrach Sastre (abril 2008)
- Francisca Alba Ramón, "Plan de prevención de riesgos laborales de una empresa constructora", dirigit per Francisco Forteza Oliver (juliol 2008)
- Francesc Canals Suau, "Guía de la documentación necesaria a aportar por parte de los principales agentes de la edificación a lo largo delproceso constructivo", dirigit per Mariano Sanz Loriente (juliol 2008)
- Sebastià Font Quetglas, "Creación de una herramienta para la consulta vía web de patologías y acciones a realizar en edificios y construcciones en general", dirigit per Sebastià Bonet Palmer i Mateu Moyà Borràs (abril 2008)
- Joaquín Cobarro Vanrell, "Levantamiento de fachadas – sistema de bajo coste", dirigit per Francisco Ponsetí Barceló (juliol 2008)
- Mónica Grau Carreras, "Estudio de la fortaleza de Isabel II (La Mola, Menorca)", dirigit per Bartolomé Martí Vidal (maig 2008)
- Unai Lilly Zuloaga, "Protocolo para la reabilitación de construcciones tradicionales mallorquinas en suelo rústico, según criterios de sostenibilidad, eficiencia energética y bajo impacto", dirigit per Joan Muñoz Gomila (setembre 2007)
- Bartomeu Adrover Barceló, "Anàlisi dels rendiments de la mà d'obra a les obres d'edificació", dirigit per Joan Muñoz Gomila (setembre 2007)

ENGINYERIA TÉCNICA INDUSTRIAL (esp. ELECTRÒNICA INDUSTRIAL)

- José Manuel Tacón Roig, "Diseño e implementación de una red de sensores basada en el protocolo ZigBee para teledetección en espacios abiertos", dirigit per Bartomeu Alorda Ladaria (setembre 2007)
- Ramon Garcias Roig, "Generació de corbes programables d'intensitat dinàmica: emulació del corrent de consum CMOS", dirigit per Bartomeu Alorda Ladaria (juliol 2008)
- Bartomeu Servera Mas, "Disseny VLSI d'un receptor d'ultrasons de baix consum en tecnologia CMOS 0.35 um", dirigit per Eugeni Isern Riutort i Miquel Roca Adrover (setembre 2007)
- Manuel Salsas Duran, "Disseny d'un sistema d'alimentació sense fils per a un sensor", dirigit per Rodrigo Picos Gayá (novembre 2007)
- Ignacio Durán Ordoñez, "Desarrollo de una plataforma de inspección visual. Aplicación a la detección de piezas", dirigit per Yolanda González Cid (octubre 2007)
- Gabriel Teruel Caro, "Disseny i automatització d'una instal.lació d'energia solar tèrmica amb seguidor solar" dirigit per Victor Martínez Moll (desembre 2007)
- Miguel Ferrer Pericás, "Estudi de viabilitat d'aprofitament energètic de biogàs a una comunitat de Nicaragua (Las Lajas)", dirigit per Andreu Moià Pol (gener 2008)
- Alejandro García Coll, "Amplificador para auriculares con válvulas de vacío" dirigit per Eugeni Isern Riutort i Miquel Roca Adrover (maig 2008)

- Pedro Benayas Vadell, "Estudi i projecte tècnic d'un hort fotovoltaic connectat a xarxa", dirigit per Debora Coll Mayor (maig 2008)
- Josep Lluis Negre Carrasco, "Reproductor MP via USB", dirigit per Eugeni Isern Riutort i Miquel Roca Adrover (maig 2008)
- Bartomeu Oliver Torres, "Disseny, implementació i programació d'una matriu d'interconnexions per automatització de mesures", dirigit per Eugeni Isern Riutort i Miquel Roca Adrover (juliol 2008)
- Guillem Mayol Capó, "Estudi i projecte d'un parc eòlic", dirigit per Eugeni García Moreno (juliol 2008)
- Luis Usandizaga Coll, "Estudio comparativo de sistemas de climatización para un hotel", dirigit per Andreu Moià Pol (juliol 2008)
- Francisco de Paula Casasnovas Barceló, "Disseny de les instal.lacions de producció elèctrica per a un habitatge aïllat amb energies renovables", dirigit per Andreu Moià Pol (juliol 2008)
- Sebastià Roca Ferrer, "Implementació i test d'un sistema de sincronització de rellotge sobre CAN", dirigit per Guillermo Rodríguez-Navas González (juliol 2008)
- Raquel Roca Durán, "Estudio y diagnosis de un aparcamiento de vehículos de uso privado", dirigit per Andreu Moià Pol (febrer 2008)

ENGINYERIA TÉCNICA TELECOMUNICACIONS (esp. TELEMÀTICA)

- Juan Baos Serrano, "Implementación de un sistema automático de seguimiento solar para placas solares", dirigit per Miquel Roca Adrover i Eugeni Isern Riutort (desembre 2007)
- Iñaki Salinas Bueno, "Prototipus d'aplicació per l'administració electrònica a la UIB. Secretaria Web", dirigit per Josep Lluis Ferrer Gomila (setembre 2007)
- Antonio Jorge Escudero Masa, "RFID en el sector turístico", dirigit per Ramon Mas Sansó (octubre 2007)
- Jaime Pons Pons, "Programació de llibreries pel control de direcció i tracció amb un microcongtrolador de 16 bits. Construcció d'un mòbil bàsic", dirigit per Bartomeu Alorda Ladaria (juliol 2008)
- Adrián Sean Amaya Moody, "Implementación de una infraestructura de clave pública utilizando software de código abierto", dirigit per Llorenç Huguet Rotger (setembre 2007)
- Martin Dario Pereyra, "Sistema d'interconsulta hospitalària", dirigit per Magdalena Payeras Capellá i Jaon Marques Faner (novembre 2007)
- Carlos Segura Orejas i Eva Ochoa de Olano Hoyos, "Implementación de una plataforma de reserva electrónica segura basada en servicios Web", dirigit per Josep Lluis Ferrer Gomila (setembre 2007)
- Jaume Sastre Tomás, "Proposta d'un laboratori de pràctiques per arquitectura de xarxes", dirigit per Ignasi Furió Caldentey (desembre 2007)
- Marcelo Medrano Looijen, "Estudio y medición de la radiación electromagnética en espacios abiertos de la UIB", dirigit per Rodrigo Picos Gayá (juny 2008)
- Jesús Martín de la Sierra Silva, "Detección soft en sistemas MIMO", dirigit per Guillem Femenias Nadal (novembre 2007)
- Tomas Font Gil, "Sistemes de localització interiors i tecnologia RFID", dirigit per Loren Carrasco Martorell (gener 2008)
- Jaime Rossiñol Pujol, "Implementación de un protocolo de compra para pagos de cantidades elevadas", dirigit per Magdalena Payeras Capellà (abril 2008)
- Carmen Castaño Camberos, "Diseño de una red de sensores RF de bajo coste, orientada a la monitorización térmica de viviendas", dirigit per Jaume Segura Fuster i Vicenç Canals Guinand

ENGINYERIA INFORMÀTICA

- Santiago Ameller Horrach, "Gestor de jocs online. El joc de truc com a exemple", dirigit per Miquel Mascaró Portells (juny 2008)
- Sebastià Rigo Riera, "Electus: una aplicació per al PIB basada en RIA", dirigit per Bartomeu Munar Pascual (octubre 2007)
- José Mariano Dellà, "Magallanes, prototipo para la navegación 3D", dirigit per Maria José Abasolo (octubre 2007)
- Antoni Lluís Mesquida Calafat, "El treball en equip en la millora dels processos software", dirigit per Antònia Mas Pichaco i Esperança Amengual Alcover (octubre 2007)
- Carlos Martínez Peña, "Estudi, desenvolupament i prova d'una aplicació MHP de creuers amb canal de retorn", dirigit per Antoni Bibiloni Coll i Miquel Mascaró Portells (desembre 2007)
- Ricardo García Noval, "Implementación de una librería destinada a la creación de aplicaciones multibiométricas", dirigit per Francisco Perales López (desembre 2007)
- Alejandro Santamarta Martínez, "Desenvolupamnet d'una aplicació per a Televisió Digital Interactiva", dirigit per Antoni Bibiloni Coll i Miquel Mascaró Portells (desembre 2007)

- Gabriel August Picó Martorell, “Reenginyeria d’un sistema rent a car via web”, dirigit per Javier Jofre González-Granda (abril 2008)
- José Luis González Pérez, “eSSIPA: supervisión SIPA”, dirigit per Miquel Mascaró Portells (juny 2008)
- Miquel Angel Rújula Gelabert, “Sistema de distribució espacial i determinació de visibilitat per al motor gràfic Nebula 2”, dirigit per Pere A. Palmer Rodríguez (setembre 2007)
- Antònia Maria Ramon Bordoy, “Protocol de correu electrònic certificat amb una tercera part de confiança verificable”, dirigit per Macià Mut Puigserver (abril 2008)
- Alberto Pozuelo Palmer, “Dotació del sistema informàtic dins l’entorn hospitalari”, dirigit pe Carlos Guerrero Tomé (abril 2008)
- José Manuel Sánchez Báez, “Sistema de información para un organismo dedicado a la formación y búsqueda de empleo”, dirigit per Javier Jofre González Granda (abril 2008)
- Marc Nadal Melià Aguiló, “Instanciació i implementació d’un format d’intercanvi per a l’experimentació amb models de rendiment”, dirigit per Catalina Lladó Matas (juliol 2008)
- David Jorro Pujol, “Harbourfid: control y gestión de accesos a puertos deportivos mediante identificación por radio frecuencia”, dirigit per Bartomeu Serra Cifre (abril 2008)
- Nieves Moro Picó, “Reingeniería e implementación de una aplicación de expansión de una cadena hotelera”, dirigit per Javier Jofre González Granda (juny 2008)
- Carlos Franco Capó, “Optimización de equipos de desarrollo en un entorno empresarial mediante Maven y herramientas de código abierto”, dirigit per Carlos Guerrero Tomé (juny 2008)
- Nicolás Milton Massetani Kupman, “Aplicación para la evaluación del canto de la perdiz roja”, dirigit per Miquel Mascaró Portells i Pere A. Palmer Rodríguez (abril 2008)
- Fco Javier de Diego y Cid, “Aibo: estudio de la plataforma y desarrollo de nuevas capacidades”, dirigit per Francisco Perales López y Gabriel Fiol Roig (juliol 2008)
- Juan Sebastián González Sureda, “Lila, un prototipo para la localización de pacientes en las salas de espera de un hospital”, dirigit per Bartomeu Serra Cifre (juliol 2008)
- María Beltrán Molla, “Cap a la gestió de la seguretat de la informació segons l’estàndard UNE-ISO/IEC 27001”, dirigit per Gabriel Fontanet Nadal (juliol 2008)

Estudia a l'Escola Politècnica Superior

Titulacions de 1er i 2on cicle:

Arquitectura Tècnica
Enginyeria Tècnica de Telecomunicacions. Especialitat en Telemàtica
Enginyeria Tècnica en Informàtica de Gestió
Enginyeria Tècnica en Informàtica de Sistemes
Enginyeria Tècnica Industrial. Especialitat en Electrònica Industrial
Matemàtiques
Enginyeria Informàtica

http://eps.uib.es

Continua els estudis a la UIB

Titulacions oficials de Postgrau

Master de matemàtiques
Màster en enginyeria electrònica
Màster en tecnologies de la informació i de les comunicacions

Doctorats

Doctorat de la UIB en Matemàtiques
Doctorat de la UIB en enginyeria electrònica
Doctorat de la UIB en tecnologies de la informació i de les comunicacions

http://cep.uib.es

Universitat de les Illes Balears

Institut de Ciències de l'Educació

Universitat de les Illes Balears

Escola Politècnica Superior