



Universidad
Zaragoza

Profesores:

Alfonso López Ruiz
Violeta Monasterio Bazán
Pablo Santafé Gabarda
Carlos Tellería Orriols
Salvador Rodríguez Sanz
Irene Sánchez Montejo

Presentación

Fundamentos de Informática

Grado en Ingeniería Electrónica y Automática – Curso académico 2026-2027



CONTACTO

2

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

3

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

4

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

5

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

6

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

7

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

CONTACTO

Alfonso López Ruiz

✉ alfonso.lopezr@unizar.es

📍 Despacho junto a salón de actos (Ada Byron)

Violeta Monasterio Bazán

✉ violeta@unizar.es

📍 Despacho C4-3-10 (Torres Quevedo) o en remoto

Carlos Tellería Orriols

✉ telleria@unizar.es

📍 Tutorías en remoto

Pablo Santafé Gabarda

✉ psantafe@unizar.es

📍 Despacho 00.070 (Betancourt)

Salvador Rodríguez Sanz

✉ srodsanz@unizar.es

📍 Laboratorio 1.01 (primera planta, Ada Byron)

Irene Sánchez Montejo

✉ irene.sanchezm@unizar.es

📍 Tutorías en remoto

Directorio UNIZAR

<https://directorio.unizar.es/#/tutoria?colectivo=PDI&codCentro=110>

► 10 prácticas.

- Programación en Java e introducción a Linux (sello 1+5).



► Organización preferentemente por parejas.

► Asistencia obligatoria y evaluación de trabajo en el aula.

- Trabajo autónomo **antes** de la sesión de prácticas: leer el guión y repasar conceptos vistos en teoría.
- Trabajo autónomo **después** de la sesión de prácticas: finalizar ejercicios y entregarlos en Moodle.

► 25% de la nota final.



UZ5/ Autoaprendizaje permanente

- Utilizar el aprendizaje de forma continuada y desarrollar estrategias de aprendizaje autónomo y flexible a lo largo y ancho de la vida para formar parte de una ciudadanía activa, motivada e integrada favoreciendo la mejora de empleo o el desarrollo personal.



TAZ-TFG-2023-2092

Implementación en FPGA del juego de la vida.

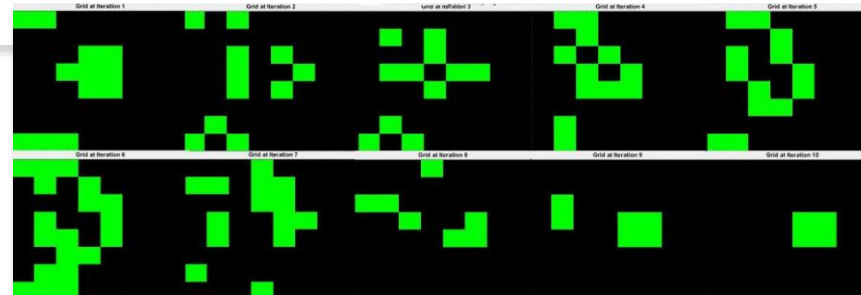
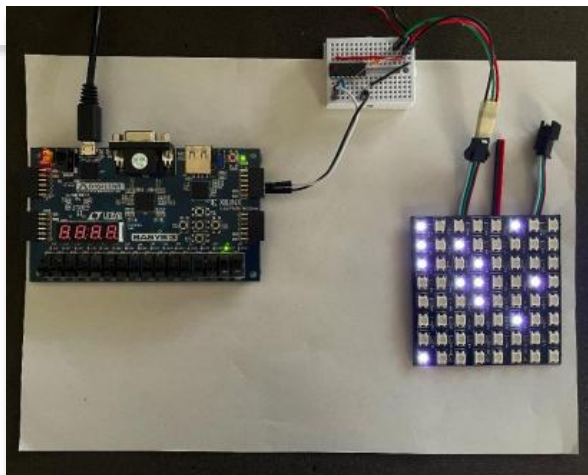
Cabrejas Escosa, Carla
Urriza Parroqué, Isidro (dir.) ; Barragán Pérez, Luis Ángel (dir.)

Universidad de Zaragoza, EINA, 2023
Departamento de Ingeniería Electrónica y Comunicaciones, Área de Tecnología Electrónica

Graduado en Ingeniería Electrónica y Automática

Resumen: El objetivo de este trabajo es diseñar, verificar e implementar en una FPGA (Field Programmable Gate Array) el autómatas celular del juego de la vida. El diseño se realiza utilizando el Lenguaje de Descripción de Hardware VHDL. Se implementa el algoritmo en Matlab para ser usado como modelo de referencia en la verificación funcional. La visualización de los patrones generados se realiza en la matriz de 8x8 LEDs WS2812. El diseño es parametrizable y los distintos bloques se conectan usando una interfaz AXI4-Stream.

-



```
begin
sig_ESTADO <= ESTADO;
EN_iterar <= '0';
GOL_MATRIX_sig <= GOL_MATRIX;
EN_tiempo <= '0';
LED_INICIO <= '0';
LED_DEBUG <= '0';
VALID_G <= '1';

case ESTADO is

when REPOSO =>
VALID_G <= '0';
LED_INICIO <= '1';
LED_DEBUG <= '1';
if(IN_INICIO = '1' and IN_INICIO_sig = '0') then
sig_ESTADO <= INICIO;
elsif(IN_DEBUG = '1' and IN_DEBUG_sig = '0') then
sig_ESTADO <= ESPERA_DEBUG;
end if;

when INICIO =>
LED_INICIO <= '1';
if(READY_G = '1') then
sig_ESTADO <= ESPERA;
end if;
```

TAZ-TFG-2025-3189

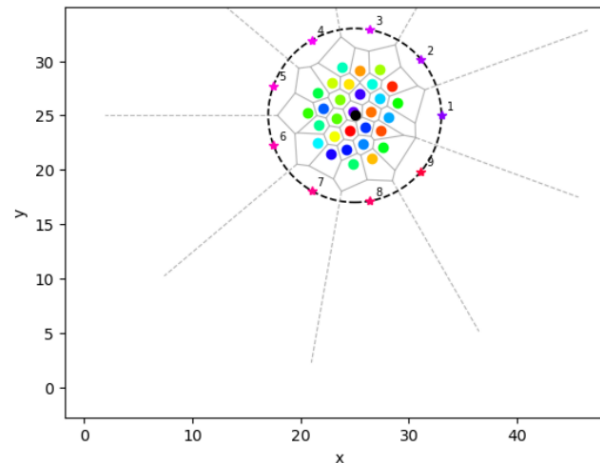
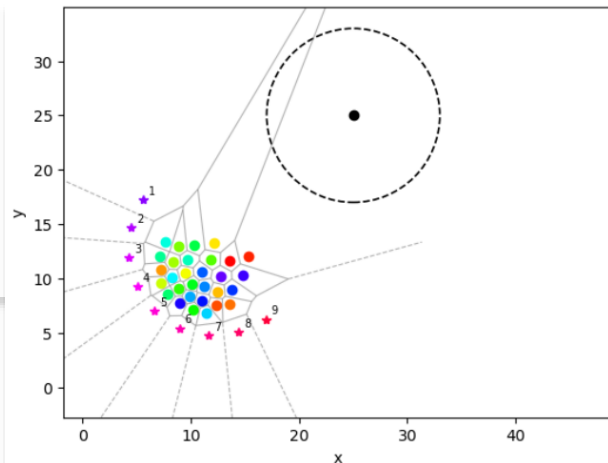
Simulación y control automático de rebaños mediante un sistema multi-agente.

Remartínez Villuendas, Leyre
Aragüés Muñoz, María del Rosario (dir.) ; Pérez Yus, Alejandro (dir.)

Universidad de Zaragoza, EINA, 2025
Departamento de Informática e Ingeniería de Sistemas, Área de Ingeniería de Sistemas y Automática

Graduado en Ingeniería Electrónica y Automática

Resumen: En este Trabajo de Fin de Grado se aborda el problema del pastoreo de ovejas mediante el uso de un grupo de robots autónomos cooperativos. El objetivo es simular el comportamiento de los perros pastores, desarrollando un sistema de control del equipo multi-robot capaz de guiar un rebaño hasta una zona objetivo y mantenerlo en su interior sin necesidad de intervención humana. Esta tarea resulta bastante compleja porque las ovejas tienden a dispersarse de manera natural, por lo que es necesario que los robots mantengan su cohesión durante todo el proceso. Como punto de partida, se ha modelado el comportamiento del rebaño utilizando técnicas de simulación de enjambres. Para ello, se ha implementado un algoritmo de agregación basado en reglas de cohesión y separación. A partir de este modelo, se ha incorporado la influencia de los robots pastores, representada mediante una fuerza de repulsión ejercida sobre las ovejas. Además, se han añadido varias mejoras respecto al algoritmo original con el objetivo de conseguir que se ajuste más a la realidad. Para realizar el control del rebaño se han implementado dos técnicas diferentes representativas del estado del arte en el pastoreo multi-robot. La primera propuesta fue presentada en un trabajo de investigación recientemente publicado en la revista IEEE Transactions on Robotics (2024). Se basa en el enfoque Outmost Push, en el que cada robot guía a la oveja visible más alejada de la región objetivo debido a que es la más



Algoritmo 2: Asignar ángulos a grupos

Input:

grupos: lista de grupos de ángulos
 r : radio de la circunferencia
 δ_{arc} : umbral mínimo de cobertura angular

Output:

Θ : lista de listas con ángulos asignados

```

1  $\Theta \leftarrow []$ 
2 foreach grupo  $g$  en grupos do
3   if  $|g| = 1$  then
4     /* Si el grupo está formado por una única oveja, añadir su
       ángulo a  $\Theta$  */
5     Añadir  $g$  a  $\Theta$ 
6   else
7     Ordenar los ángulos del grupo:  $g \leftarrow \text{sort}(g)$ 
8     Calcular ángulo inicial  $\theta_{ini}$  y final  $\theta_{fin}$ 
9     Calcular arco:  $\delta \leftarrow (\theta_{fin} - \theta_{ini}) \bmod 2\pi$ 
10    Calcular longitud del arco proyectado sobre la circunferencia:  $L \leftarrow \delta \cdot r$ 
11    if  $L < \delta_{arc}$  then
12      /* Si el arco que forman las ovejas del grupo sobre la
         circunferencia es menor al umbral  $\delta_{arc}$ , asignar un
         único ángulo medio */
13      Calcular ángulo medio:  $\theta_{med} \leftarrow (\theta_{ini} + \delta/2) \bmod 2\pi$ 
14      Añadir  $[\theta_{med}]$  a  $\Theta$ 
15    else
16      /* Si el arco es grande, asignar varios ángulos
         equiespaciados */
17      Calcular número de segmentos necesario para cubrir arco:  $s \leftarrow \lfloor L/\delta_{arc} \rfloor$ 
18      Generar  $s + 1$  ángulos equidistantes entre  $\theta_{ini}$  y  $\theta_{fin}$ 
19      Añadir lista de ángulos generados a  $\Theta$ 
20 return  $\Theta$ 

```

TAZ-TFG-2025-3203

Desarrollo de sistema de aviónica para un cohete de competición de ALUnizar

Ubieta Tabuenca, Daniel
Casas Nebra, Roberto José (dir.)

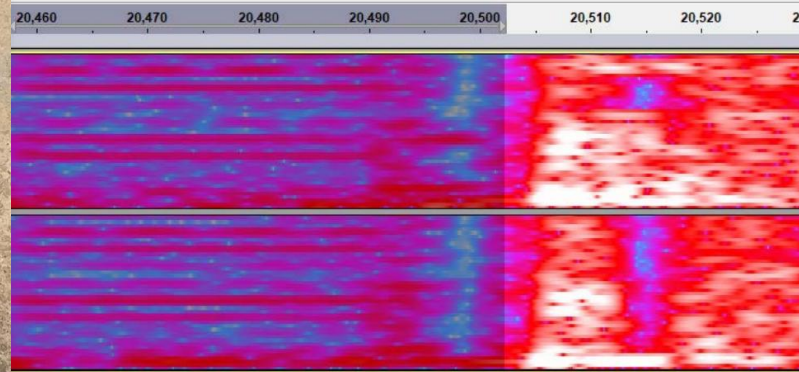
Universidad de Zaragoza, EINA, 2025
Departamento de Ingeniería Electrónica y Comunicaciones, Área de Tecnología Electrónica

Graduado en Ingeniería Electrónica y Automática

Resumen: El proyecto tiene como objetivo el diseño y la fabricación de una PCB para el control y la recolección de datos de un cohete construido con fines académicos. Este trabajo ha sido propuesto por ALUnizar, un equipo de la Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza. ALUnizar se dedica a la fabricación de cohetes para competiciones académicas y científicas. Uno de los componentes más relevantes de un cohete es la aviónica, responsable tanto del despliegue del paracaídas como de la recopilación de parámetros clave como altura, velocidad, dirección y lugar de aterrizaje, entre otros. ALUnizar requiere que la placa tenga las siguientes funcionalidades: • Realización de medidas inerciales durante todas las fases del vuelo. • Transmisión por radio de las medidas tomadas por los sensores. Estos datos serán recibidos por una estación terrestre, cuyo desarrollo queda fuera del alcance de este TFG. • Control de dos salidas de corriente a que controlarán la detonación de dos cargas de pólvora en los momentos precisos para el despliegue de los paracaídas. Además la placa será la responsable de estimar el mejor momento para el despliegue de estos. Para conseguir esto ALUnizar ha estipulado que la placa debe tener: • Barómetro. • Giroscopio. • Magnetómetro. • GPS. • Zumbador. • Modulo LoRa. • Dos salidas de corriente para los paracaídas. Para cumplir con los requerimientos de ALUnizar, la placa se pretende usar en la competición EuRoC (European Rocket Competition) que es uno de los requisitos necesarios para poder participar en esta competición. Durante el desarrollo del TFG, la placa se ha diseñado con el objetivo de mantener compatibilidad ante posibles modificaciones futuras. El diseño se ha realizado con el objetivo de mantener compatibilidad ante posibles modificaciones de EuRoC de 2025 ya que el cohete aún no está fabricado pero se diseñará para cumplir estas reglas. El sistema de aviónica ha sido diseñado para cumplir los requisitos impuestos por ALUnizar. a continuación se ha creado un prototipo de la placa de circuito impreso (PCB) que se utilizará para el desarrollo del sistema de aviónica.



Tipo de



```
if(activacionDROGUE()){
    siguienteEstado = drogueDesplegado;
    digitalWrite(DROGUE,HIGH);
    millisCuandoSeApagaDROGUE = millis() + tiempoDeActivacionCargas;
}
break;
case drogueDesplegado:
    if(millis() >= millisCuandoSeApagaDROGUE){
        digitalWrite(DROGUE,LOW);
    }
}
```

```
#define cantidadAlturas 128
float arrayAlturas[cantidadAlturas]; //Guardar lo equivalente a 1.28
segundos de datos
int punteroArrayAlturas = 0;
bool activacionDROGUE(){ //Cuando se llegue al punto de Apogeo
    arrayAlturas[punteroArrayAlturas] =
    ps.pressureToAltitudeMeters(ps.readPressureMillibars());
    bool diferenciaAltura = arrayAlturas[punteroArrayAlturas] -
    arrayAlturas[(punteroArrayAlturas-1) % cantidadAlturas] <= 4.905*(
    (float)(periodoRapido*128)/1000.0 )^2;
    bool estamosSubiendo = arrayAlturas[punteroArrayAlturas] -
    arrayAlturas[(punteroArrayAlturas-1) % cantidadAlturas] >= 0;
    bool noEstamosEnDespegue = ((dato.acc[0]^2 + dato.acc[1]^2 +
    dato.acc[2]^2) < 0.5^2) && (arrayAlturas[punteroArrayAlturas] >
    alturaDespegue + 50.0);

    punteroArrayAlturas = (punteroArrayAlturas++) % cantidadAlturas;
    return diferenciaAltura && estamosSubiendo && noEstamosEnDespegue;
}
```

C++

```
int total = 0;

for (int i = 0; i < 10; i++) {
    if (i % 2 == 0) {
        total = total + i;
    }
    else {
        total = total - i;
    }
}
```

50M repeticiones -> **0.029** s

Python

```
total = 0

for i in range(10):
    if i % 2 == 0:
        total = total + i
    else:
        total = total - i
```

50M repeticiones -> **7.6** s (266x)

VHDL

```
process
    variable total : integer := 0;
begin
    for i in 0 to 9 loop
        if (i mod 2 = 0) then
            total := total + i;
        else
            total := total - i;
        end if;
    end loop;

    wait;
end process;
```

Not available ☹️

Java

```
int total = 0;

for (int i = 0; i < 10; i++) {
    if (i % 2 == 0) {
        total = total + i;
    }
    else {
        total = total - i;
    }
}
```

50M repeticiones -> **0.062** s (2.2x)

