industriales
etsii

Escuela Técnica Superior de Ingeniería Industrial

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

Mejora de equipamiento para la caracterización de células fotovoltaicas orgánicas

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA ELÉCTRICA

Autor: **Rodrigo Félix Peral**
Director: José Abad López
Codirector: Javier Padilla Martínez

Cartagena, Septiembre de 2015

Universidad Politécnica de Cartagena

AGRADECIMIENTOS

Quiero agradecer este trabajo a mis tutores José Abad y Javier Padilla por su tiempo, su ayuda, sus ideas y su confianza.

A Charlotte por su ayuda y compañía durante este largo año de trabajo, además de sus conocimientos de Word y estilo. Gracias por ser tan paciente.

A Adrián por acercarme a la idea que acabó siendo el núcleo de este proyecto: automatizarlo con Arduino.

A Israel por su paciencia, dedicación y ayuda sin la cual no hubiera podido hacer este trabajo.

Por supuesto a mi familia y amigos por mantenerme con vida.

INDICE

1. INTRODUCCIÓN

El creciente aumento de las necesidades energéticas exige el desarrollo de nuevas fuentes de energía. Quizás una de las opciones más interesantes sea la explotación de energías de tipo renovable, que nos permitan aprovechar fuentes de energía de un inmenso potencial, como puede ser la radiación solar incidente en la tierra, aun desplazadas por otras formas más conocidas e utilizadas.

Será en 1945 cuando se empiece a desarrollar esta posibilidad mediante el desarrollo de las células fotovoltaicas basadas en semiconductores como el silicio por Champin, Fuller y Pearson en los laboratorios Bell (1). Esta tecnología ha avanzado de forma significativa hasta llegar en la actualidad a unos rendimientos en la conversión de energía entre el 15-20% para células de silicio monocristalinas. Sin embargo, un problema que presenta este tipo de tecnología es que requiere de un complejo y costoso proceso de fabricación.

Por lo que el desarrollo de nuevos materiales y nuevos métodos de fabricación es una necesidad para el abaratamiento de la energía. La utilización de elementos orgánicos parece una solución factible para este problema aparte de añadir otras posibilidades derivadas de sus interesantes propiedades como puede ser su fácil integración arquitectónica.

Estas células fabricadas a base de polímeros semiconductores han llegado a arrojar valores de eficiencia del 5%(2) así como añadir nuevas posibilidades al ser parcialmente transparentes además de flexibles.

Sin embargo la tecnología basada en semiconductores orgánicos tienen a su vez el inconveniente de degradarse de forma significativa frente a factores ambientales (humedad, oxigeno, temperatura) y su exposición a la radiación solar, mientras otras tecnologías, basadas en componentes inorgánicos, presentan una mayor resistencia a la degradación. Esta deficiente estabilidad es debida entre otros factores a foto-oxidación, cambios de morfología y degradación química tanto de los electrodos como de las diferentes capas que componen el dispositivo. La degradación será un factor fundamental y muy a tener en cuenta a la hora de caracterizar nuestras células (3).

La gráfica 1 muestra algunos comportamientos habitualmente encontrados para la evolución de la eficiencia (PCE:power conversión efficiency) de células sometidas a irradiación.

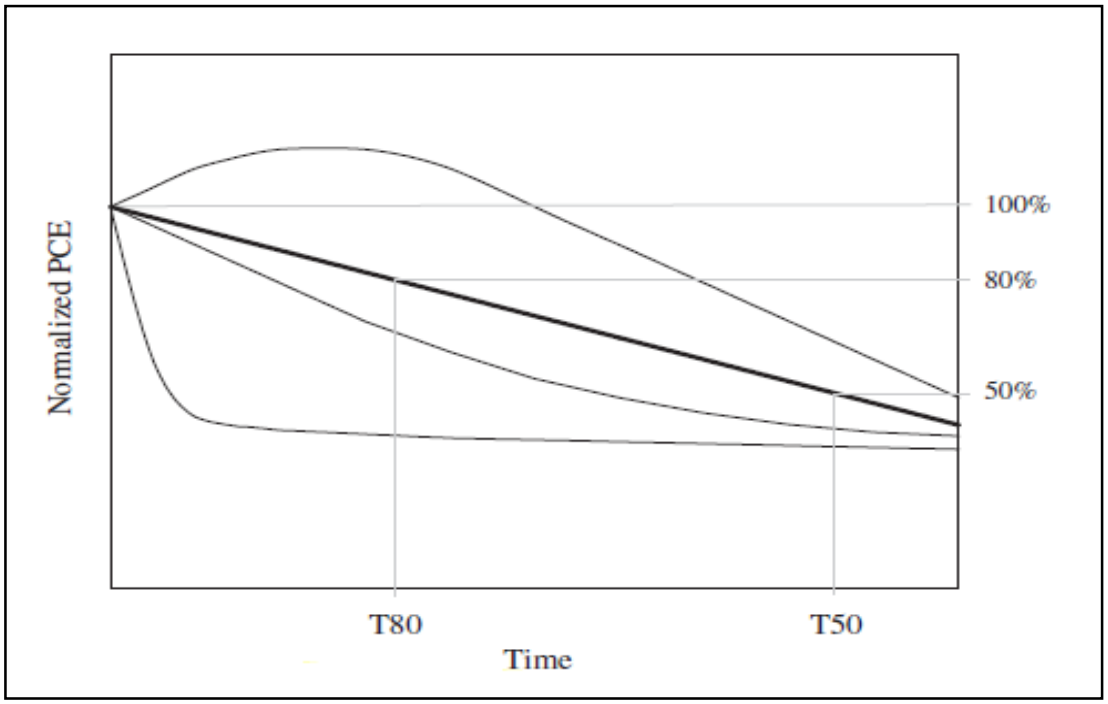


Figura 1. Gráfico de degradación temporal de células orgánicas (3).

1.1. CARACTERIZACIÓN

En el desarrollo de nuevos tipos de células, en los que se varía tanto su composición como los métodos de fabricación, se necesita una forma estándar de caracterizar dichos dispositivos para así valorar los resultados obtenidos y poderlos comparar con las diferentes tecnologías de fabricación y materiales.

La caracterización que realizaremos será entorno a los parámetros habituales de las células fotovoltaicas: corriente de cortocircuito, voltaje de circuito abierto, eficiencia en la conversión de energía, factor de llenado, resistencia serie, resistencia paralelo, etc. Estos parámetros se obtendrán de la característica I-V (Corriente-Tensión) obtenida mediante ensayos para cada célula en concreto y en unas condiciones ambientales y de iluminaciones también concretas, en este caso proporcionadas por un simulador solar con un espectro AM1.5 (1000W/m2).

La curva I-V será el objeto de nuestro estudio. A continuación se muestra en la figura 2 una curva I-V característica.

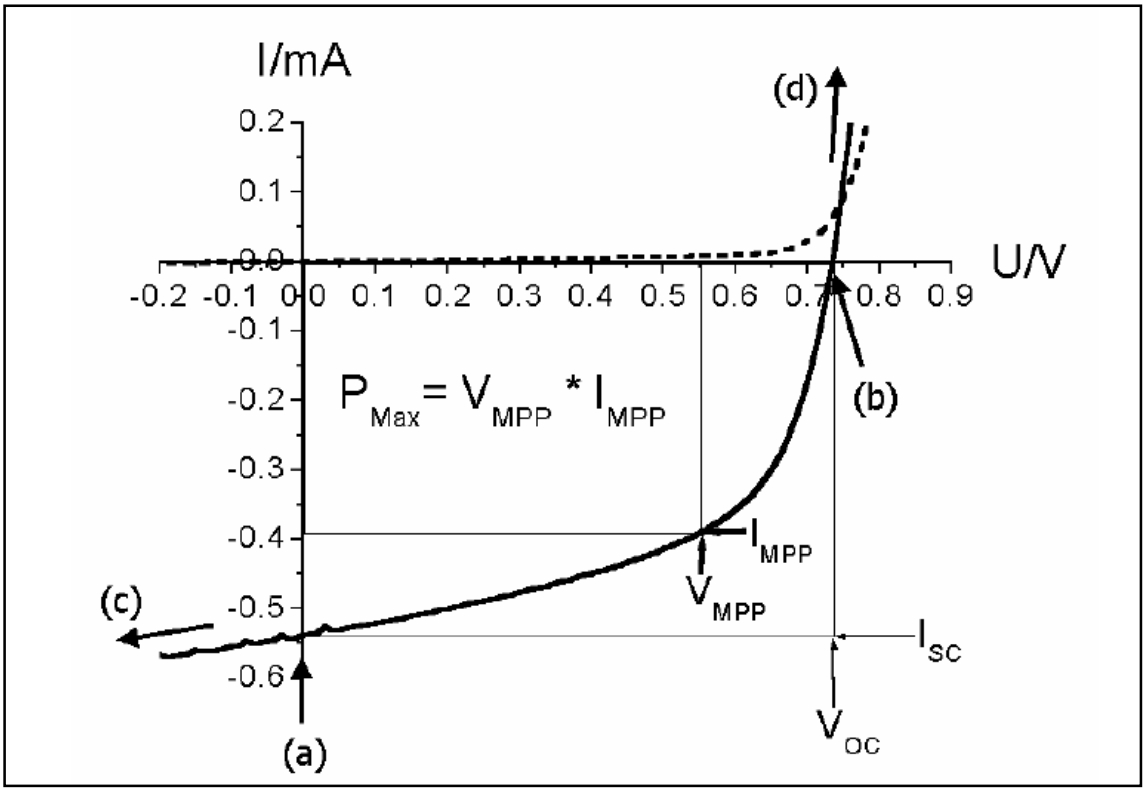


Figura 2. Curva IV y diodo de una célula fotovoltaica

Como podemos ver la curva es similar a la de un diodo (curva en línea discontinua) debido a las características de las células (unión de semiconductores p-n) pero desplazada, de forma tal que en el cuarto cuadrante podemos observar la aparición de una corriente inversa que utilizaremos para la generación de energía.

Uno de los principales parámetros a definir es el Factor de Forma o llenado (Fill Factor-FF). Mediante este parámetro podemos dar cuenta de la calidad de una célula. Esto lo consigue comparando el área formada por la curva experimental con la que tendría en caso de funcionar en su punto de máxima potencia, el cual se puede obtener con el producto de las tensiones y corrientes obtenidas. Así compararíamos con los valores máximos de tensión en Circuito abierto (V_{oc}) y corriente de cortocircuito (I_{sc}) respectivamente (puntos de corte de la curva con los ejes). Así obtenemos una idea de cuánto se asemeja nuestra célula a una fuente de corriente constante a una tensión constante que suministraría la potencia máxima.

$$FF = \frac{P_{máx}}{I_{sc}V_{oc}} = \frac{(I * V)_{máx}}{I_{sc}V_{oc}}$$

Otro aspecto interesante es la eficiencia de la célula. Para calcularla solo tenemos que relacionar la energía que ésta recibe en forma de luz (P_{light}) con la potencia que genera:

$$\eta = \frac{P_{máx}}{P_{light}} = \frac{I_{sc} * V_{oc} * FF}{P_{light}}$$

Esto muestra que hay algunos elementos internos que imposibilitan que la célula trabaje de forma ideal. Estos elementos habrá que tenerlos en cuenta a la hora de formar el circuito equivalente.

Para poder hallar un equivalente que podamos analizar y comparar de nuestra célula tendremos que tener en cuenta lo visto hasta ahora de forma que tenemos:

- Fuente de Corriente (célula iluminada)
- Diodo debido a la unión PN
- Resistencia serie (R_s)
- Resistencia paralelo (R_{sh})

Este circuito será similar a una fuente de corriente que no es capaz de proporcionar una tensión constante. Quedará de la siguiente forma:

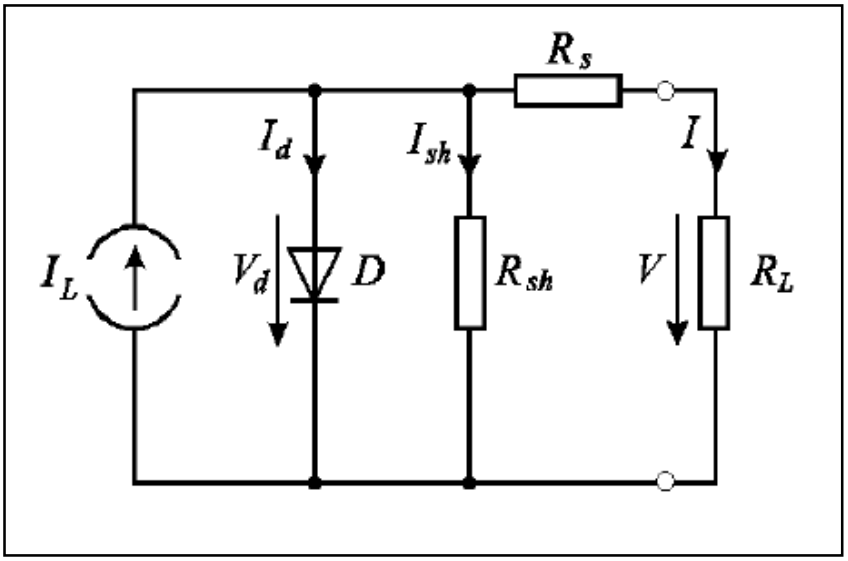


Figura 3. Circuito equivalente de una célula fotovoltaica

Este circuito contempla los diferentes estadios en los que se dan las pérdidas durante el funcionamiento normal de una célula, así pues describiremos cada elemento:

- I_l representa la corriente generada debido a la iluminación.
- R_{sh} es debida a la recombinación de los portadores de carga dentro de la unión semiconductora (pares electrón-hueco). Suele ser la mayor de las resistencias y representa pérdidas internas de la célula.
- R_s también contempla cierta parte de recombinación aunque es generalmente asociada a las pérdidas que supone tener una corriente en un medio, incluidos los contactos.

Los valores de R_{sh} y R_s se pueden obtener de forma analítica a partir de la curva I-V, siendo representada en el primer cuadrante, tomando la pendiente de ésta en los puntos de corte con ambos ejes como podemos ver en la figura siguiente:

- D, o el diodo ideal, da cuenta de la naturaleza de los semiconductores y su comportamiento no lineal, el cual se puede apreciar en la curva I-V.
- V es la tensión que genera la célula, siendo su valor máximo V_{oc}. Este valor es dependiente de la carga y por eso será una variable que nosotros controlemos durante la caracterización de la célula, obteniendo así los distintos puntos de la curva I-V.

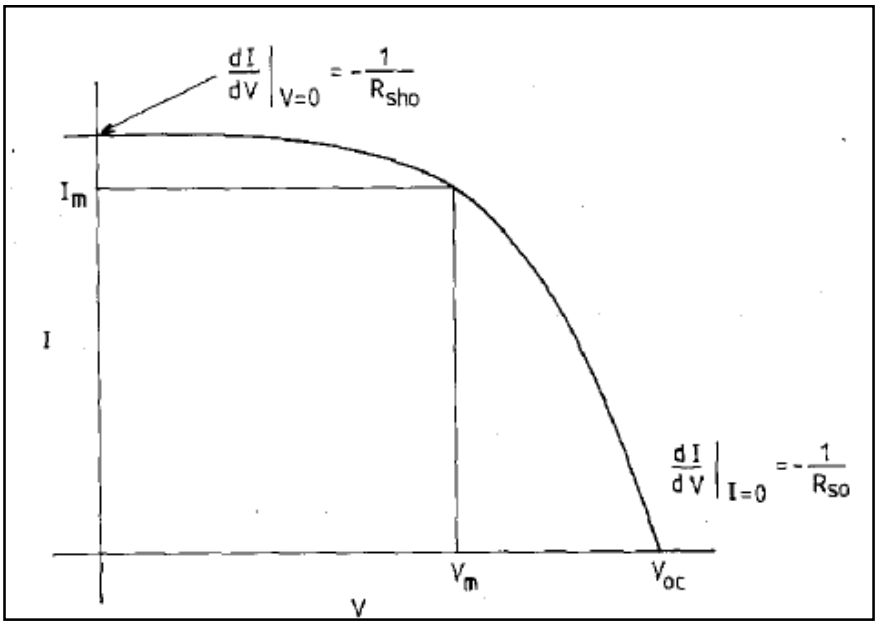


Figura 4. Localización de Rs y Rp en la curva de una célula fotovoltaica (Primer cuadrante)

También es posible encontrar modelos más complejos que añaden un segundo diodo y una capacidad que son útiles para el estudio del plano completo, nosotros solo tendremos en cuenta el comportamiento en el cuarto cuadrante, por lo que nos es más que suficiente.

Si realizáramos un análisis del circuito obtenido podríamos desarrollar un modelo matemático que nos permita conocer las relaciones entre los distintos parámetros expuestos. Así el desarrollo que podemos encontrar en la tesis desarrollada por Klaus (4) deviene en:

$$I = \frac{I_L - \frac{V}{R_{sh}}}{1 + \frac{R_s}{R_{sh}}} - \frac{I_0}{1 + \frac{R_s}{R_{sh}}} * \left(e^{\frac{V - IR_s}{nKT/q}} - 1\right)$$

Podemos pues concluir que es fundamental el conocimiento de estos parámetros de forma que a lo largo de una serie de ensayos podamos conocer el comportamiento de las células (2)(3).

1.2. LABVIEW

Para realizar dicho análisis será necesaria la implementación de un programa capaz de tomar las medidas correspondientes, en este caso la característica I-V registrando la corriente de la célula mediante la variación del potencial en la misma (Variación de la carga), y calcular con éstas los parámetros característicos. El programa seleccionado en este caso es LABVIEW, un programa desarrollado por National Instruments para la medida y control de procesos.

LABVIEW es una plataforma de programación gráfica que ayuda a escalar desde el diseño hasta pruebas y sistemas pequeños hasta grandes sistemas. Además ofrece una incomparable integración con hardware de adquisición de datos tanto de National Instruments como de terceros, bibliotecas de procesamiento de señales y controles de interfaz de usuario construidos con propósitos específicos para visualización de datos de medidas. Es también importante su capacidad de automatizar procesos en los que intervienen múltiples medidas y toma de decisiones basadas en éstas.

La forma funcional básica de LABVIEW es la de programación gráfica, esto se realiza mediante bloques que desarrollan distintas operaciones o funciones que pueden ser interconectados mediante “hilos” para representar el tránsito de datos dentro del mismo. Así se permite la rápida visualización del funcionamiento de un programa así como su implementación o mejora sin necesidad de sumergirse en un código abstracto y confuso (6).

1.3. INDICE DE ANTECEDENTES Y EXPECTATIVAS

El desarrollo del proyecto se centra fundamentalmente en conseguir implementar un proceso de medida lo más coherente posible con los estándares existentes para la caracterización de las células (1), de forma que puedan ser de utilidad los datos tomados durante los ensayos realizados en el laboratorio.

Es interesante introducir el estado del sistema de caracterización para analizar las mejoras que pretenden implementarse en éste.

El sistema cuenta con un simulador solar de espectro AM1.5 (1000W/m^2) que sirve de fuente de luz, además tenemos una fuente de tensión regulable (KEITHLEY model 230 Programable Voltage_Source) que sirve de carga variable, un amperímetro de precisión (KEITHLEY Model 6514) para tomar las lecturas de corriente y un ordenador con el que interconectar y gobernar los distintos elementos mediante la plataforma LABVIEW.

Hasta ahora la toma de datos se hacía situando las células bajo el simulador solar y conectándolas a sendos aparatos de medida mediante unas pinzas de tipo cocodrilo a los contactos. A continuación se tenía que ejecutar un programa de labview que iniciaba un ciclo de medidas según unos parámetros fijados por el usuario (número de pasos, Tensión inicial, etc.)

Los problemas que podemos encontrar en el sistema anterior tienen que ver, sobre todo, con lo rudimentario del proceso: cada sustrato contiene 4 células que tienen que ser medidas independientemente, de forma que para ensayar cada una hay que recolocar los cocodrilos en los contactos correspondientes. Esto también imposibilita la realización de un ensayo de degradación, lo cual exigiría la presencia continua de una persona en el laboratorio que iniciara el experimento, teniendo en cuenta que solo podría ser realizable a una de las células del sustrato.

Las ideas principales son el desarrollo de una base que permita conectar la célula a los aparatos de medida de forma cómoda, similar a las bases comerciales propuestas por FOM technologies (fig. 6), entre otras, que permitan conectar estas a una serie de relés para posibilitar la toma de datos de las 4 células de un sustrato de forma consecutiva sin necesidad de variar los contactos. Todo esto debe estar acompañado de una mejora en el software disponible, ampliando las capacidades de los programas disponibles en labview.

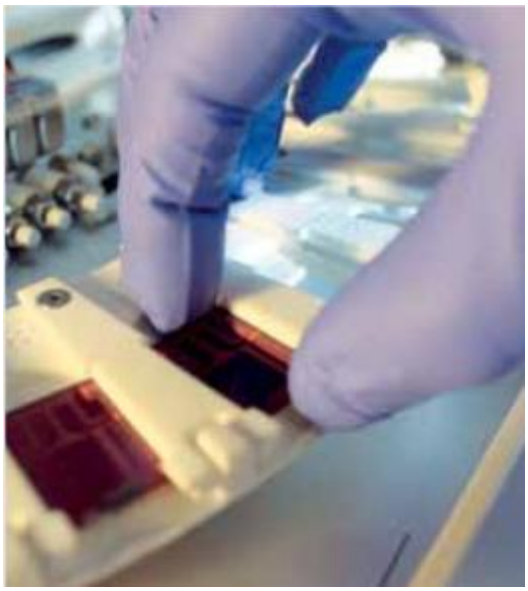


Figura 6. The FOM Device Testing Carousel

Otros elementos como la automatización del procesos de toma de datos como temperatura y humedad así como la integración de todo el sistema mediante un dispositivo microcontrolador como arduino son tenidas en cuenta según los progresos que se puedan ir realizando.

2. ANTECEDENTES

Pasamos a describir la instalación precedente. Al describirla pretendemos presentar el método de caracterización poniendo en relieve la importancia del trabajo realizado como forma de subsanar las carencias existentes.

2.1. HARDWARE

Como ya hemos mencionado, la caracterización de células se realiza mediante el trazado de sus curvas de respuesta o curvas características (curvas IV). En este proceso, que puede realizarse bajo diferentes condiciones ambientales (temperatura, humedad, iluminación, etc.), se desarrolla mediante la aplicación de un potencial variable a la célula obteniendo sus valores de intensidad de corriente correspondientes. Esta relación Tensión-Corriente (característica IV) nos permitirá caracterizar la célula.

Para poder aplicar el potencial y obtener la intensidad de corriente la célula ha de conectarse a los terminales de una fuente de tensión y un electrómetro. Esta conexión se realiza gracias a dos pinzas o cocodrilos, una de las cuales apoya en el sustrato de la célula (ITO) y otra en el contacto metalico (Al oAg) correspondiente.

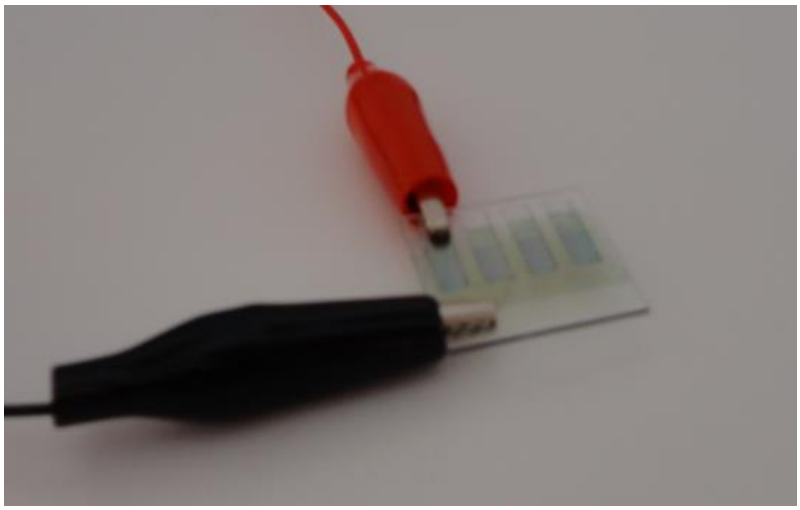


Figura 7. Conexión de una célula durante el ensayo.

Así podemos identificar 2 momentos clave en el desarrollo del experimento: la aplicación de un potencial y la medición de la corriente resultante.

De estos se encargarán:

- KEITHLEY Model 230 Programable Voltage Source (Fuente de Tensión)

El modelo 230 de Keithley es una fuente de tensión programable con un rango de tensiones entre 199.95mV y 101V. Cuenta con la capacidad de elegir entre tres límites de corriente de 2mA, 20mA y 100mA. Esta fuente también puede ser gobernada mediante un interface GPIB.

(IEEE-488) (7).

- KEITHLEY Model 6514 System Electrometer (electrómetro)

El modelo 6514 de Keithley es un electrómetro capaz de medir voltaje, corriente, resistencia y carga eléctrica. Su rango de medida, para corrientes, es de +-21mA con una precisión de hasta pA. También es compatible con sistemas de comunicación GPIB (8).

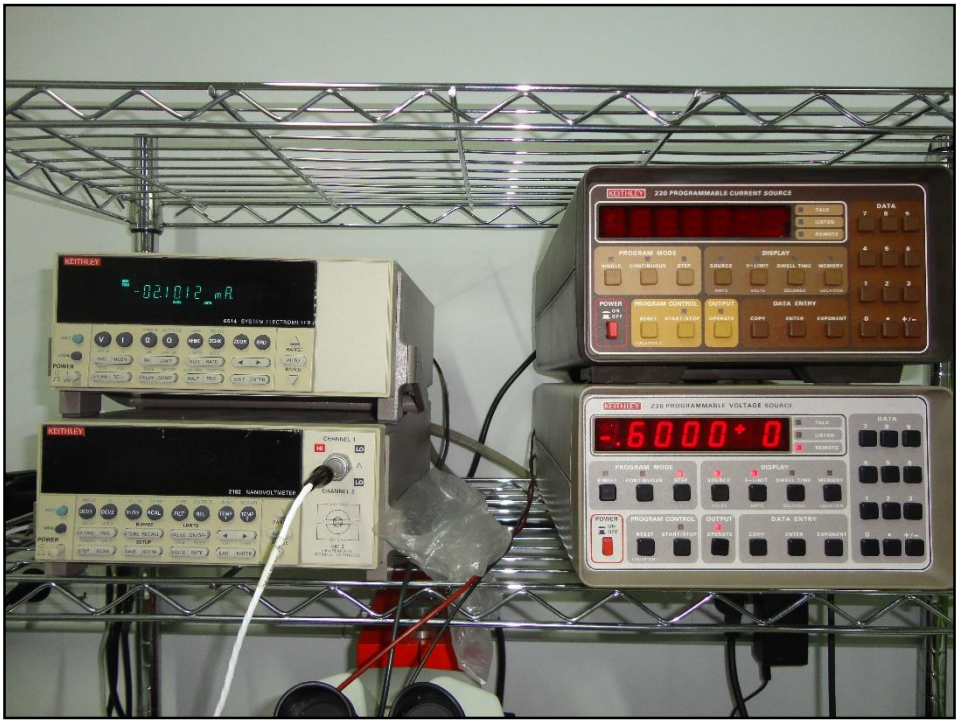


Figura 8. Fuente de tensión y electrometro Keithley

Como la finalidad de la experiencia es obtener una característica es necesario tomar varias lecturas. Este proceso será guiado por un ordenador que disponga de la plataforma de control LABVIEW. Gracias a este sistema será el ordenador el encargado de comunicar los valores de tensión de cada medida a la fuente y de registrar la corriente asociada a dicho valor de tensión.

La comunicación entre la fuente de tensión y el electrómetro se llevará a acabo mediante una tarjeta GPIB (General-Purpose Instrumentation Bus). Esta tarjeta es capaz de gestionar el proceso de comunicación entre el ordenador, el cual se encarga de controlar todo el proceso mediante un programa implementado en LABVIEW, la fuente de tensión y el electrómetro. En nuestro caso disponemos de una tarjeta GPIB de National Instruments. Esta es reconocida

automáticamente por el software VISA de National Instruments quien se encarga de instalar los controladores pertinentes, quedando así lista para su uso con LABVIEW.

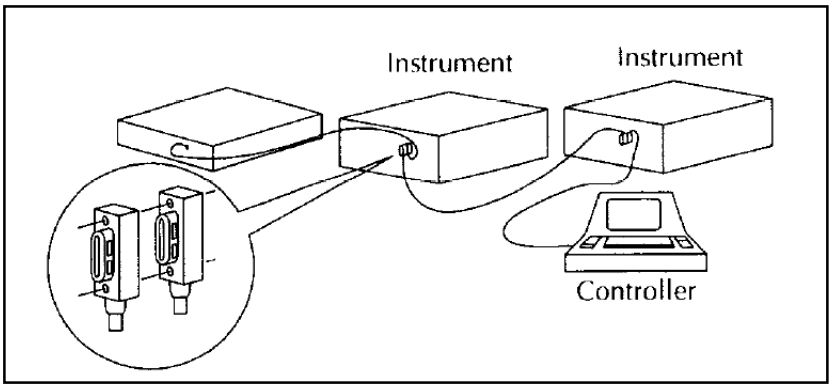


Figura 9. Conexión de instrumentos GPIB

Para poder realizar experimentos de forma controlada se dispone de un simulador solar ABET 2000.

*Figura 10.*Simulador solar ABET 2000.

Para el registro y almacenamiento de los datos contamos con un ordenador básico con Windows XP así como la versión 8.6 de LABVIEW. Éste ejecuta un programa de LABVIEW que se encarga de ejecutar un ciclo de medición del cual obtenemos una característica IV así como los valores de tensión de circuito abierto, corriente de cortocircuito, factor de llenado y eficiencia.

Este proceso de toma de datos tiene asociado una serie de problemas. El principal quizás sea que se necesita de una persona que controle la adquisición de datos, encargándose de conectar cada una de las células existentes en uno o varios sustratos.

Este problema lleva aparejada la imposibilidad de realizar experimentos de degradación, pues la toma de datos de forma regular durante largos periodos de tiempo; una persona debe encargarse de iniciar el programa así como de cambiar la célula a probar si así fuese necesario.

Otro problema importante es el conexionado de las células mediante las pinzas cocodrilo debido a la existencia de variabilidad en los datos por un mal contacto entre las sondas y los contactos de la célula, llegando incluso a dañar los contactos metálicos o rayar el ITO; esto también impide la correcta colocación de la célula respecto al simulador solar, no siendo ésta perpendicular ni manteniendo una altura constante.

2.2. SOFTWARE

Para el control del proceso de medida se había desarrollado un software en la plataforma de control LABVIEW. Este software se encargaba del trazado de curvas IV y posterior cálculo de los parámetros más importantes como son la tensión de vacío, la corriente de cortocircuito, el factor de llenado y la eficiencia de la célula.

El programa presentaba una pantalla principal en la que ingresar los principales parámetros del ensayo. La ejecución del mismo es bastante sencilla e intuitiva; únicamente hay que rellenar los campos de entrada e indicar a LABVIEW que queremos iniciar el experimento.

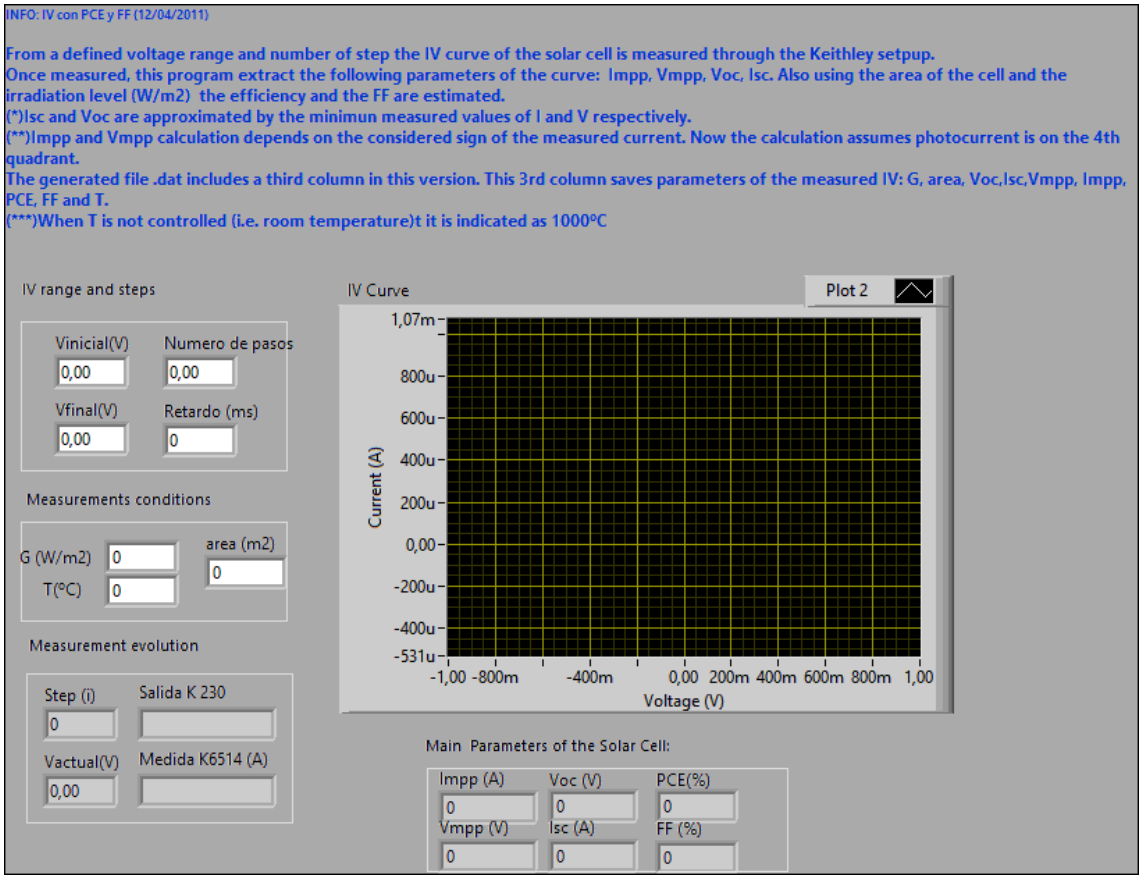


Figura 11. Pantalla principal del programa precedente

Como podemos observar tenemos como entradas parámetros ambientales y constructivos de la célula. Estos son:

- Irradiación (G): energía luminosa incidente en la célula en W/m².
- Temperatura: temperatura a la que se realiza el experimento en grados Centígrados.
- Área: superficie de la célula expuesta a la luz en m².

El resto de parámetros están relacionados con el desarrollo del experimento:

- Vinicial: tensión mínima o de partida para el experimento en Voltios.
- Vfinal: tensión máxima o final del experimento en Voltios.
- Número de Pasos: número de puntos de la curva a realizar.

Los parámetros de salida podemos dividirlos en dos grupos.

Los primeros muestran las lecturas instantáneas de los instrumentos:

- Step: Paso en el que se encuentra el experimento del total de pasos requeridos.
- Salida k 230: Tensión suministrada por la fuente de tensión en Voltios.
- Vactual: Tensión aplicada para este paso del experimento.
- Medida K6514: Corriente registrada por el electrómetro en Amperios.

El resto solo estarán disponibles una vez finalizado el experimento debido a la necesidad de utilizar todos los datos obtenidos. Éstos son:

- Impp: Corriente de máxima potencia. Muestra el valor de la corriente para el punto de máxima potencia en Amperios.
- Vmpp: Tensión de máxima potencia. Muestra el valor de la tensión para el punto de máxima potencia en Voltios.
- Voc: Tensión de circuito abierto en Voltios.
- Isc: Corriente de cortocircuito en Amperios.
- PCE: eficiencia de la célula en tanto por ciento.
- FF: Factor de llenado de la célula en tanto por ciento.

Por último disponemos de un trazador de gráficos (IV curve) que mostrará la curva resultante una vez finalizado el experimento permitiendo conocer su forma.

El programa detrás de este interfaz responde a una lógica bastante sencilla: a partir del número de pasos y las tensiones de inicio y fin calcula la tensión a aplicar en cada ejecución, después envía dicho valor de tensión al K230 (fuente de tensión), para por último leer el valor de corriente que envía el K6514 (amperímetro). Los valores obtenidos se guardan emparejados en una matriz para su posterior utilización. Este proceso se repite tantas veces como se haya especificado mediante el número de pasos.

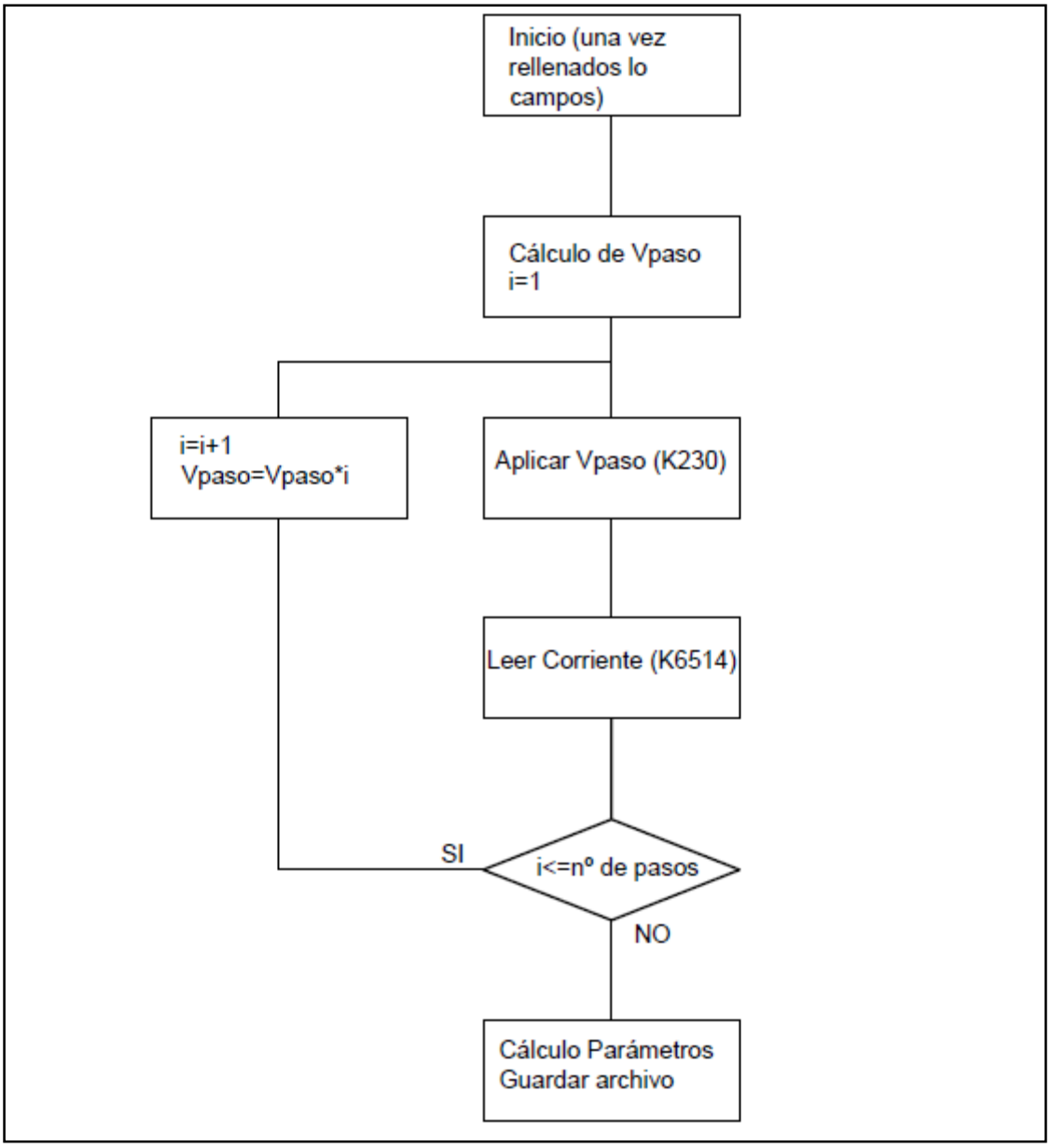


Figura 12. Flujograma del programa precedente

Para el cálculo del rango de tensión correspondiente a cada paso del ciclo se aplica la siguiente expresión:

$$V_{paso} = \frac{V_{final} - V_{inicial}}{número\ de\ pasos}$$

Este valor se actualiza en cada iteración. Siendo "i" la iteración en la que nos encontramos:

$$V_{paso} = V_{paso} * i$$

Desde la perspectiva de LABVIEW el programa se construye mediante un diagrama de bloques como el siguiente:

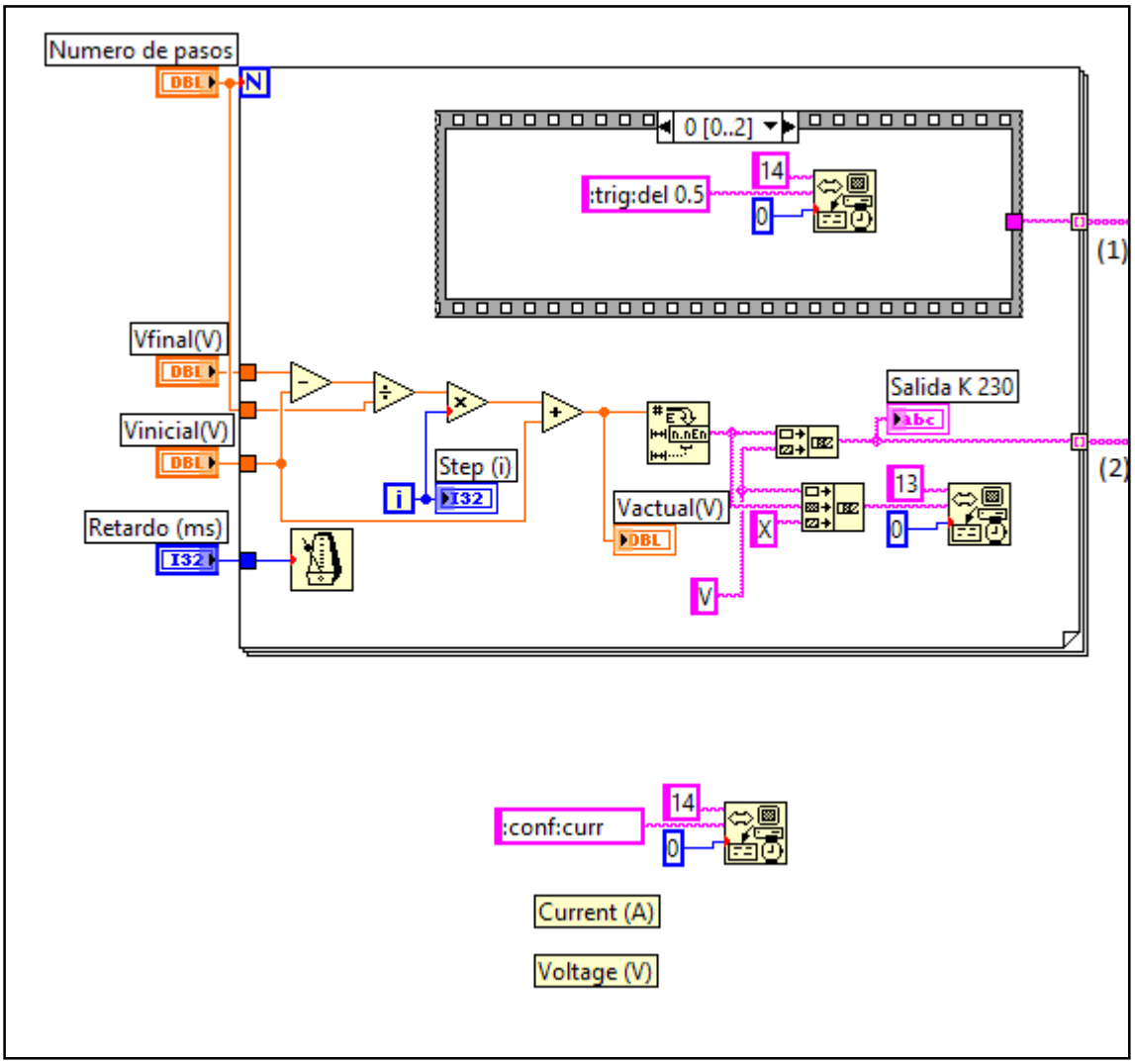


Figura 13. Diagrama de bloques del programa precedente en Labview (I)

Esta primera parte sería la encargada de la toma de medidas y su almacenamiento.

La siguiente se encargaría de la manipulación de dichos datos para obtener la eficiencia, el FF, etc. Así como su posterior almacenamiento en un archivo txt.

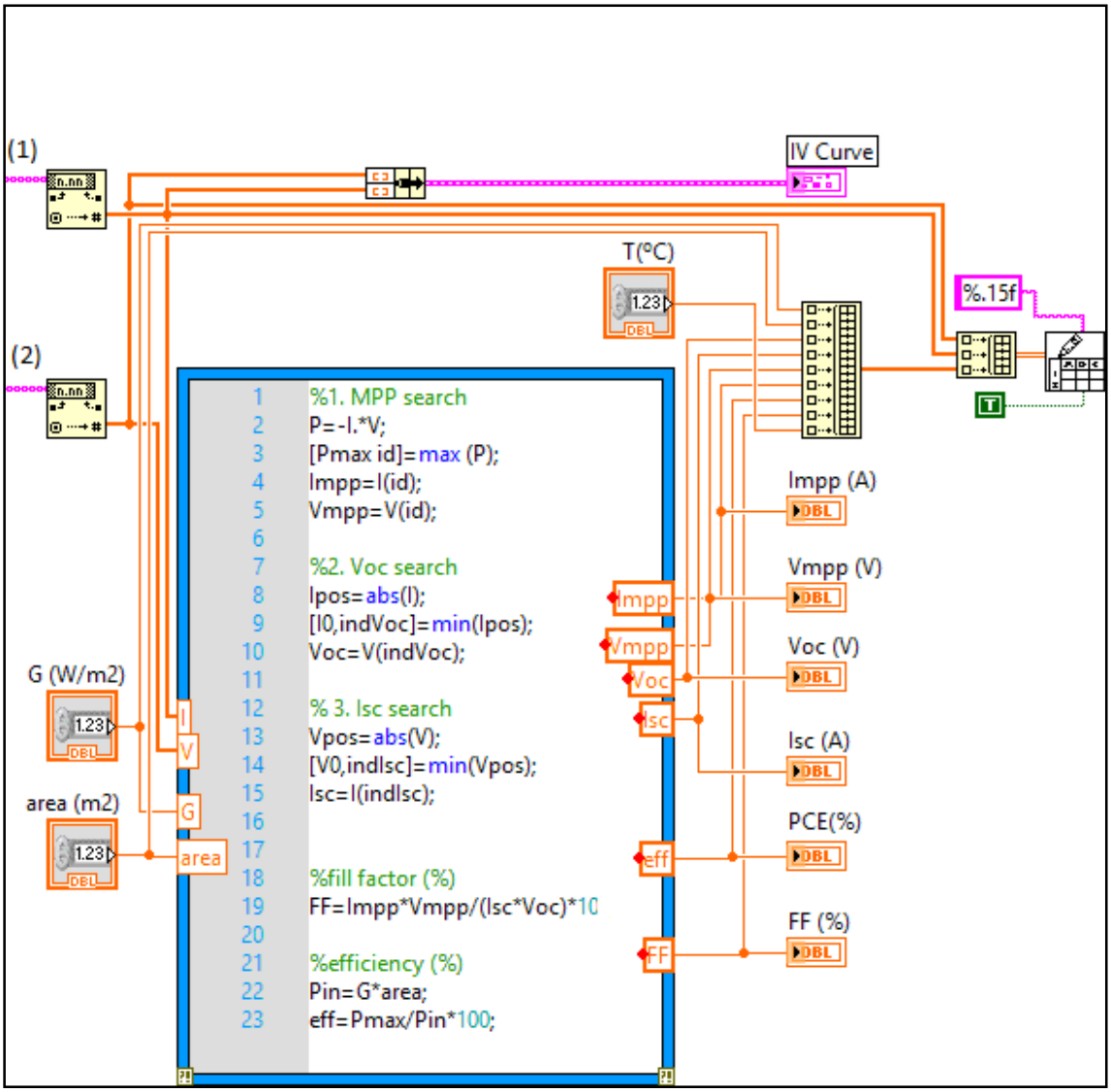


Figura 14. Diagrama de bloques del programa precedente en Labview (II)

3. SOFTWARE: LABVIEW Y ARDUINO.

El entorno en el que desarrollaremos el software necesario para alcanzar nuestros objetivos está compuesto por LABVIEW y ARDUINO. LABVIEW nos proporcionará la base encargada del control del proceso y posterior gestión de los datos obtenidos; mientras, ARDUINO se encargará de gestionar parte de los recursos hardware necesarios para la automatización del proceso de medición como son relés y sensores.

Esto será posible gracias a las características particulares de ARDUINO, el cual, al ser una plataforma "libre", nos permite usar librerías para una fácil inclusión en LABVIEW. A continuación pasaremos a describir el proceso de instalación y configuración de los distintos elementos software así como algunas de las funciones más básicas y representativas de esta unión especial entre LABVIEW y ARDUINO.

En nuestro caso contamos con la versión 2011 de LABVIEW.

Para comenzar necesitaremos tener instalado un software de LABVIEW (habría que contemplar las compatibilidades con las librerías y/o complementos de ARDUINO para LABVIEW). Este software tiene que tener ciertos complementos de LABVIEW como son:

- National Instruments VISA.
- VI Package Manager.
- National Instruments Measurement & Automation
- Software ARDUINO.

3.1. VISA

Se trata de la arquitectura software virtual para instrumentos (Virtual Instrument Software Architecture). Este software es un estándar para la configuración, control y programación de instrumentos. Es capaz de gestionar interfaces software USB, como GPIB, etc.

A modo de ejemplo podemos resaltar "VISA WRITE" como una de las funciones fundamentales que nos aporta esta plataforma para el control de instrumentos; nos permitirá enviar información al instrumento en cuestión.

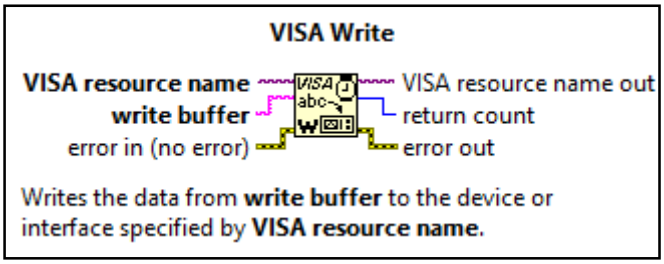


Figura 15. Función Labview "VISA Write"

3.2. VI PACKAGE MANAGER

Es una plataforma que te permite crear, compartir y encontrar diferentes bloques de herramientas para utilizar con LABVIEW. En nuestro caso es quien nos va a proveer de la aplicación de ARDUINO para LABVIEW. Simplemente instalándolo y ejecutándolo obtendremos de forma inmediata una lista de todos los complementos disponibles. En nuestro caso necesitaremos "LabView Interface for Arduino".

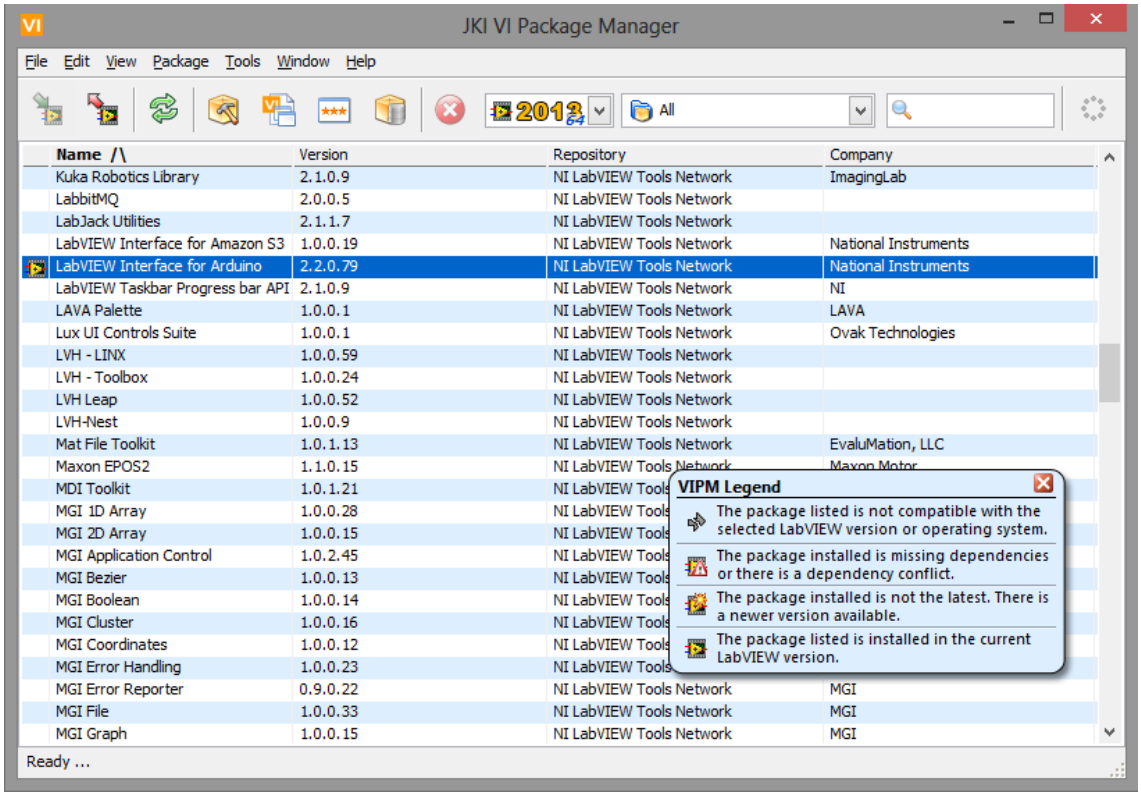


Figura 16. Pantallazo del programa JKI VI Package Manager

Si seleccionamos el complemento deseado se nos abrirá una pantalla que nos permitirá la instalación del mismo.

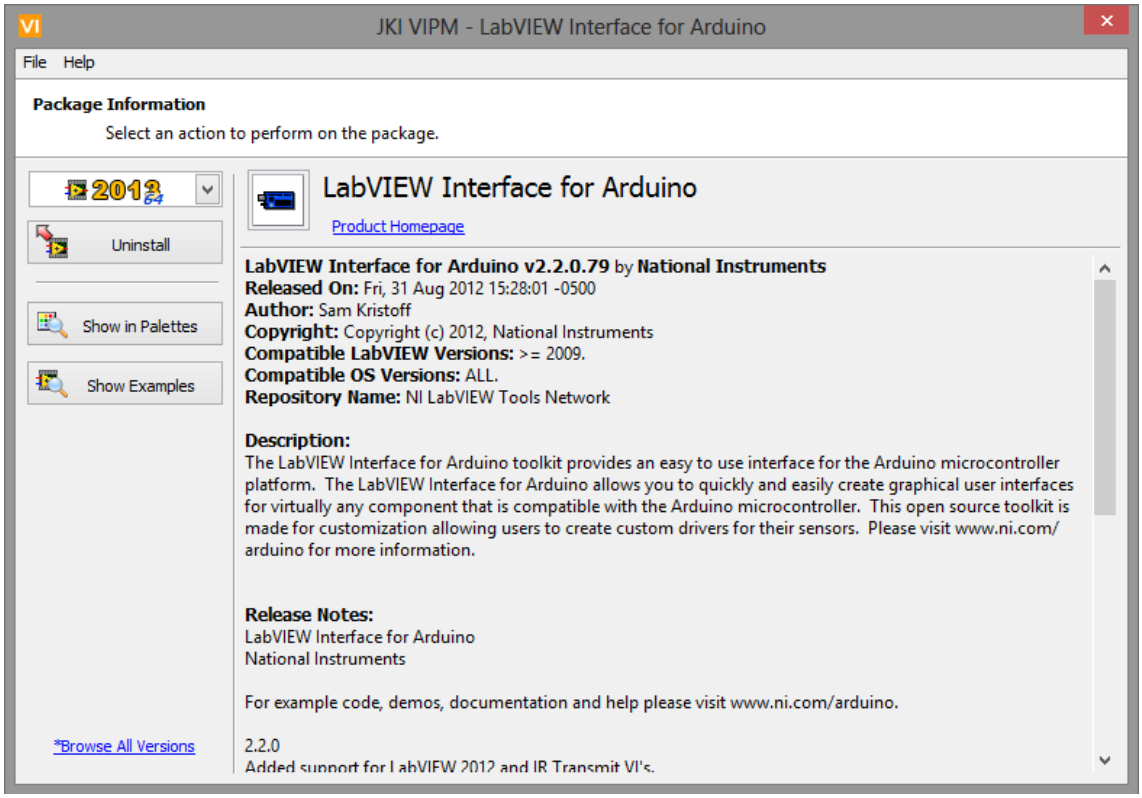


Figura 17. Pantallazo de la instalación de "LabVIEW Interface for Arduino"

Una vez instalado podremos descubrir que las funciones disponibles en nuestra paleta de LABVIEW cuentan con un nuevo campo llamado "Arduino".

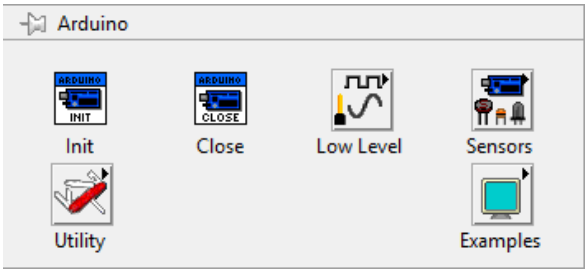


Figura 18. Paleta de herramientas Arduino para LABVEIW

En éste podremos encontrara todas las funciones necesarias para la comunicación con el hardware de ARDUINO. A modo de ejemplo podemos enseñar "init". Esta función inicializa el hardware de ARDUINO (que en nuestro caso está conectado vía USB) desde un programa en LABVIEW.

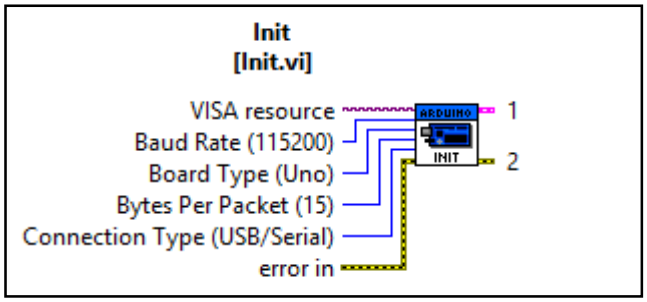


Figura 19. Función "Inicio" para Arduino en LABVIEW

3.3. ARDUINO

El proceso normal de funcionamiento de ARDUINO responde al siguiente método:

1. Desarrollamos el código/programa que resuelva nuestra aplicación
2. Cargamos el código en nuestro hardware ARDUINO
3. Trabajamos con él de forma autónoma.

El software, en este caso libre, con el que se desarrolla el código y con el que se carga el programa en el dispositivo hardware es "Arduino Software"(https://www.arduino.cc/en/Main/Software). Podemos ver su apariencia en la figura siguente:

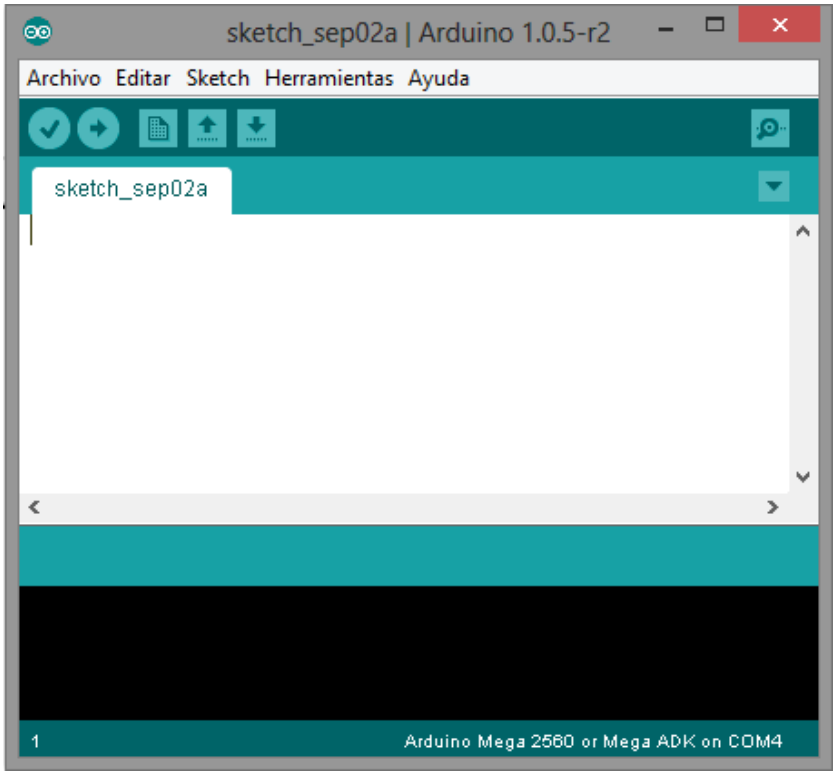


Figura 20. "Arduino Software"

En nuestro caso el proceso cambia un poco debido a que queremos que sea un programa desarrollado en LABVIEW el que controle nuestro ARDUINO vía USB. Para ello tendremos que cargar un programa llamado “LIFA_BASE” en el hardware ARDUINO que tendremos disponible una vez instalado “LabView Interface for Arduino” de VI Package Manager.

Lo que tendremos que hacer será abrir el archivo “LIFA_BASE” para cargarlo en nuestro ARDUINO. Para ello tendremos que localizar el puerto de comunicación en el que se encuentra (COM1, COM2, COM3,…), así como nuestro modelo hardware, en nuestro caso un Arduino MEGA 2560.

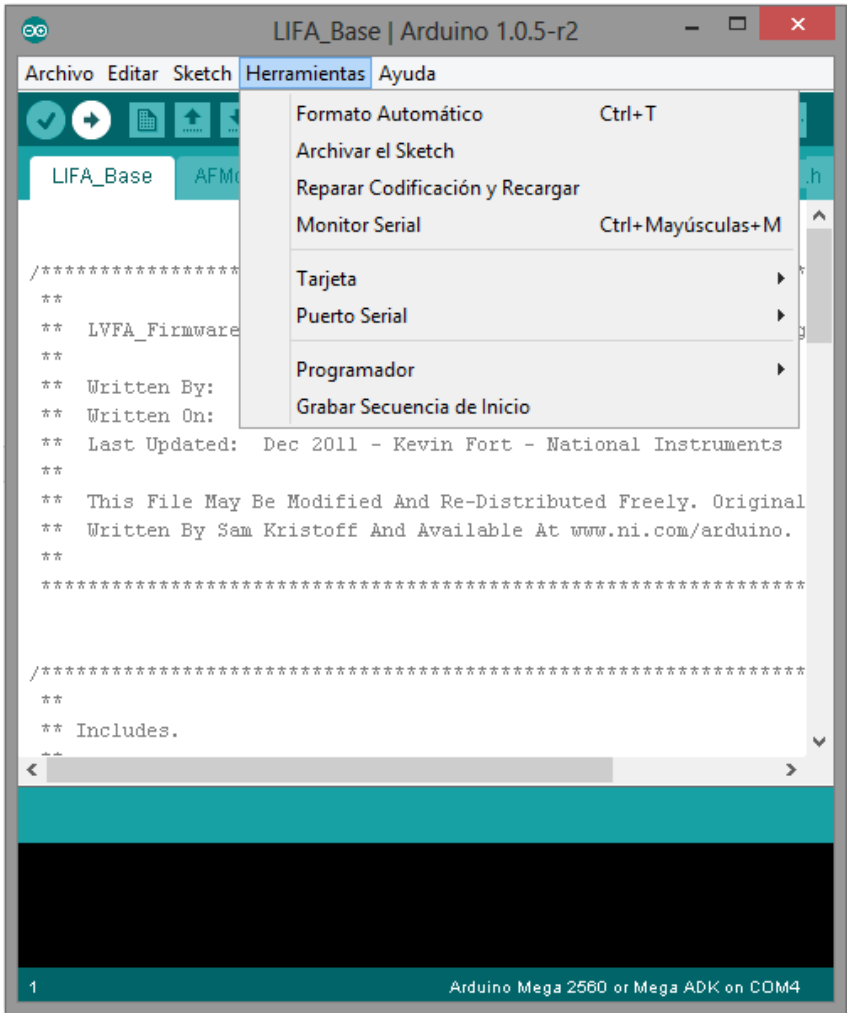


Figura 21. Pantallazo durante la carga de LIFA_Base en Arduino

Será en el menú de herramientas donde seleccionaremos nuestra “Tarjeta” (Arduino Mega 2560 en nuestro caso) y “Puerto Serial”. Una vez seleccionado bastará con seleccionar el botón de “Cargar” para traspasar nuestro código al dispositivo Arduino.

Siguiendo estos paso tendremos nuestro ARDUINO listo para ser utilizado con las funciones antes mostradas para LABVIEW.

4. SOLUCIÓN/IMPLEMENTACIÓN HARDWARE.

En este punto planteamos el objetivo fundamental del trabajo, el cual trata sobre la sistematización y automatización de la toma de datos/caracterización de las células solares fabricadas en el laboratorio. Esto implica la posibilidad de tomar medidas en las 4 células de que consta cada sustrato, la evaluación de las condiciones ambientales en cada ensayo (temperatura y humedad), así como posibilitar la realización de ensayos de degradación.

Con la vista puesta en implementar estas mejoras hemos decidido enfrentarlo desde el punto de vista del hardware en tres etapas o hitos:

1. Desarrollo de un soporte/base para los sustratos que permita tener acceso a las 4 células que lo componen.
2. Implementación de un sistema de relés que permita seleccionar qué célula perteneciente a que sustrato queremos medir/caracterizar.
3. Implementación de un sistema de medida de las condiciones ambientales.

A continuación desarrollaremos estos campos.

4.1. BASE PARA SUSTRATO

Es quizás el hito más importante del proyecto debido a que sin la implementación de esta mejora no sería posible ejecutar la siguiente etapa del trabajo. La idea principal en construir un dispositivo capaz recibir y acomodar el sustrato y permitir el acceso a los distintos contactos eléctricos de cada célula como son el ITO común a las 4 células del sustrato y el contacto de aluminio de cada célula.

Uno de los principales problemas presentes en esta etapa es el grosor de los contactos de metálicos. Éstos son crecidos sobre la célula solar en un proceso de evaporización térmica en vacío obteniendo finalmente unos contactos que pueden tener desde unos pocos nanómetros hasta una micra de espesor. Esto los hace especialmente vulnerables a ser rayados impidiendo así un buen contacto eléctrico, incluso llegando a inutilizarlos impidiendo la reproducibilidad de los experimentos. La dificultad fundamental estriba en encontrar un método en el que se ejerza cierta presión sobre los contactos, garantizando una correcta continuidad eléctrica sin dañarlos.

Nuestra idea es crear un receptáculo para los sustratos que sea capaz de cerrar los distintos circuitos eléctricos, mantener el sustrato en una posición fija y horizontal respecto al simulador solar, permita el paso de la luz a las células y sea fiable y cómodo de utilizar.

Para este fin pensamos en desarrollar una base de material plástico consistente en un hueco, capaz de recibir el sustrato, en el que se encuentre ubicados unos contactos metálicos, que cerraran nuestro circuito, y una tapa que presione el sustrato contra los contactos, de forma que el circuito quede perfectamente cerrado.

Hasta obtener un resultado satisfactorio hemos tenido que crear diferentes prototipos que, mediante ensayo-error, nos permitieran obtener los conocimientos necesarios para la adopción de una solución satisfactoria. En concreto hemos desarrollado 4 prototipos, de los cuales mostraremos los 3 primeros para poder apreciar el avance:

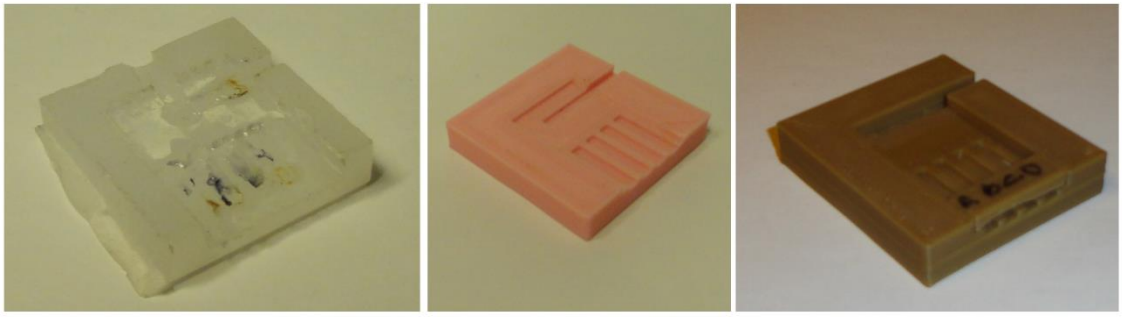


Figura 22. Prototipos de base para sustrato

La primera aproximación fue una base de teflón mecanizada de forma manual con una fresa de 3 ejes que había disponible en el laboratorio. El principal problema que surge es la falta de experiencia en los procesos de mecanizado, además la situación y forma de fijación de los contactos no era aún evidente (se prueba con pegarlos directamente a la base obteniendo un resultado insuficiente debido a que estos se despegaban y/o no hacían buen contacto).

Esta primera experiencia es fundamental en nuestra progresión. A partir de ella desarrollamos un nuevo diseño en el que podemos ubicar de forma fija los contactos. Además abandonamos la idea del mecanizado y la sustituimos por la impresión 3D. Este método nos permite una rápida evolución gracias a la posibilidad de modificar directamente el prototipo anterior en un programa de diseño, su impresión y su puesta a prueba.

Otro aspecto que también avanza es el sistema de cierre. Para los primeros prototipos se utilizaron unas pinzas para documentos. Estas pinzas tenían tres problemas: ejercían una presión fija y muy grande (llegando incluso a partir algún sustrato), además de ocupar mucho espacio alrededor de la base (no hubiera sido posible ubicar 4 sustratos bajo el simulador), y hacían muy difícil colocar el sustrato y cerrar el dispositivo después.

Como conclusión mencionar que la modelización 3D es la herramienta ideal para este tipo de trabajos pues nos ha permitido poner en marcha diferentes prototipos y modificarlos in situ una vez encontramos sus carencias.

Una vez adquirida esta experiencia pudimos desarrollar un último prototipo:

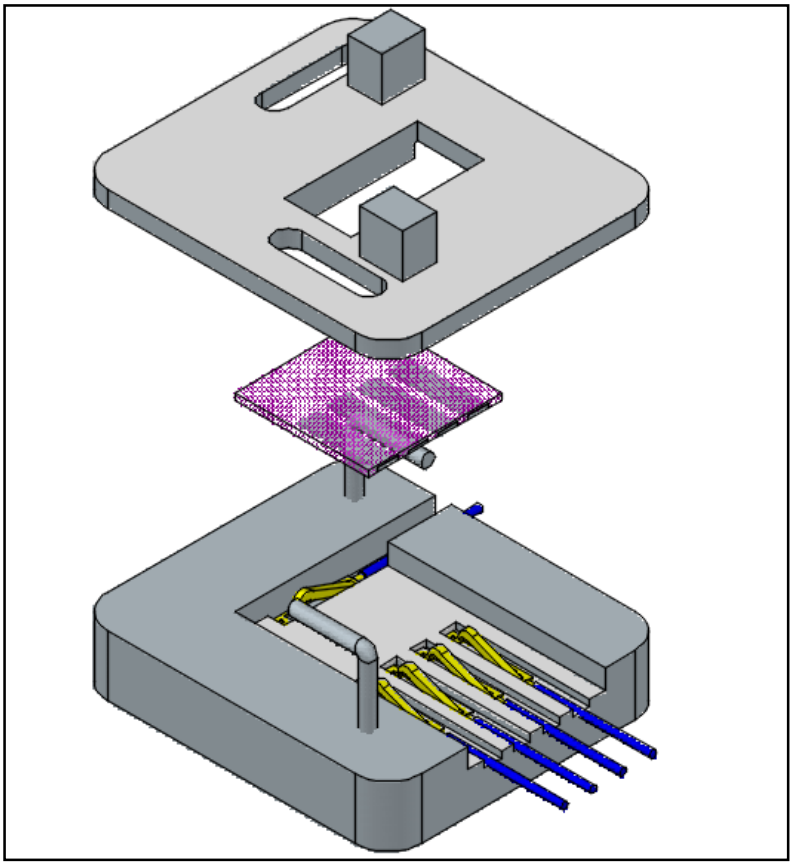


Figura 23. Dibujo 3D en AUTOCAD del prototipo definitivo

Consta de una base sólida con un hueco, en el que se encuentran unas patillas encargadas de cerrar el circuito con los contactos de metálicos, sobre la que descansará el sustrato. Sobre la base se situará una tapa que cumplirá la doble función de presionar el sustrato contra los contactos manteniéndola inmóvil, y dejar pasar la luz a las células.

Este sistema cubre todas las necesidades que habíamos planteado:

- Cumple como soporte para el sustrato.
- Permite tener acceso eléctrico a los 4 contactos de aluminio y al común(ITO)
- No daña los contactos de aluminio debido a que las patillas ceden lo suficiente como para presionar sin rallar el contacto y el sustrato no cambia su posición una vez colocado.
- Da acceso de la luz a las células.

Para su implementación hemos decidido utilizar la tecnología de impresión 3D. El proceso empieza con el diseño en la plataforma AUTOCAD. Nuestro prototipo consta de las siguientes piezas:

4.1.1. Base

La base es el elemento sobre el que descansará el sustrato, además alojará los huecos en los que se insertarán un par de cáncamos encargados de fijar la tapadera. La base es la parte más complicada de desarrollar debido a los problemas que podemos encontrar durante la impresión, siendo el principal el warping (combado de la pieza en las esquinas debido a una mala adherencia del plástico ABS). Para solucionar este problema redondeamos las esquinas de la base y disminuimos el factor de relleno de la pieza (la pieza no es totalmente maciza, su parte interior está estructurada de forma hexagonal como un panal de abeja). En nuestro caso el factor de relleno será del 20%.

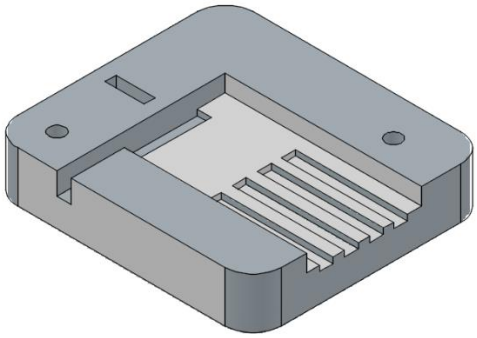


Figura 24. Base para sustrato

4.1.2. Puente

El puente será la pieza encargada de mantener los contactos de la base en inmóviles en los carriles. Debido a que la impresión de las piezas se hace por capas estas quedan estriadas a lo largo de su superficie vertical, gracias a lo cual el puente podrá encajar simplemente presionando sobre la base.

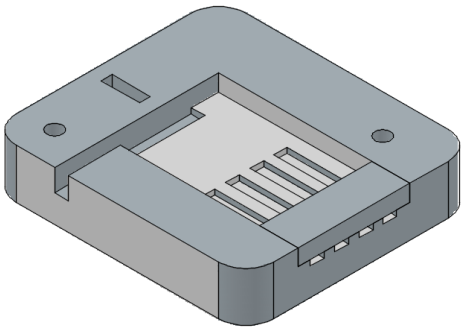


Figura 25. Base para sustrato con puente (comparación modelo-impresión)

4.1.3. Tapadera

La principal función de la tapadera es fijar el sustrato en el hueco de la base y presionarlo contra los contactos. La tapa tendrá que dejar pasar la luz por lo que se le añadirá una ventana en la parte coincidente con las células del sustrato.

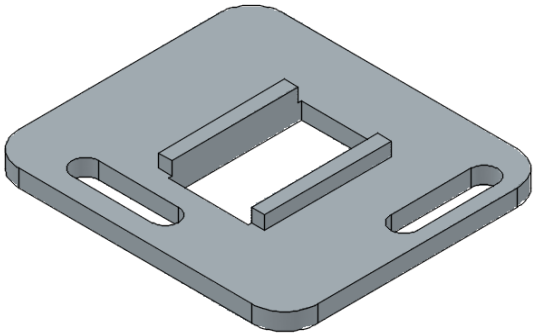


Figura 26. Parte inferior de la tapa

Como podemos ver ésta tiene dos salientes que se encargarán de hacer presión sobre el sustrato. Este punto es importante debido a la variabilidad de los espesores de los sustratos que en nuestro caso han sido de 0,7-1mm.

4.1.4. Tacos

Los tacos son dos cubos que posteriormente serán adheridos a la tapa para permitir que los cáncamos cierren la tapa sobre la base. Los tacos no se pueden imprimir directamente sobre la tapa pues esto supondría hacer la pieza prácticamente entera en “voladizo”. Resulta más sencillo imprimirlos por separado y adherirlos mojando la cara que hará contacto con la tapa en acetona, de forma que el ABS se ablanda y permite el pegado.

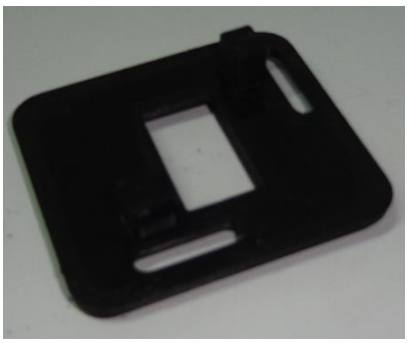


Figura 27. Tapa con tacos pegados

AUTOCAD también nos permitirá exportar nuestro diseño a un archivo de texto plano (.stl) con el que podremos trabajar en un software específico para impresión. Para obtener nuestro archivo .stl simplemente tendremos que buscar la opción “exportar”, donde seleccionaremos “otras opciones” de las cuales usaremos “litografía(.stl)”.

Esta opción nos permite seleccionar los objetos que deseamos guardar como imprimibles de todo nuestro espacio de trabajo en AUTOCAD.

Para la impresión hemos utilizado filamento de Poliacrilonitrilo Butadieno Estireno (ABS L1) de 3mm y negro opaco (9). Este plástico se imprime a una temperatura de 245ºC. De entre los plásticos disponibles (ABS y PLA) es el más resistente a condiciones de irradiación solar así como el más duro y con mejores características para tratamiento en post-producción. También hay que tener en cuenta la posibilidad de ser pegado a si mismo utilizando acetona, esto nos posibilitará implementar de forma fácil nuestro diseño.

Por último queda mencionar los contactos. Los terminales que hemos utilizado a modo de contacto son unas pastillas que vienen dispuestas por pares sobre una estructura de plástico. Estas patillas deben ser separadas y soldadas a un cable. Una vez montadas el "puente" será el encargado de, presionando los cables, mantenerlas en su sitio.

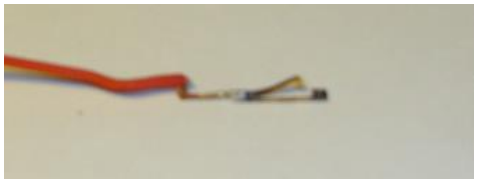


Figura 28. Patilla soldada a su cable

4.2. ARDUINO MEGA

Arduino MEGA 2560 es una placa con microcontrolador basada en ATMega2560. Consta de 54 pins digitales que pueden comportarse como entradas y salidas (15 de los cuales pueden usarse como salidas PWM), 16 entradas analógicas, etc. En la siguiente tabla podemos ver sus principales características (Tabla 1).

Tabla 1. *Características Ardiuno (10)*

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Este dispositivo puede ser programado por ArduinoSoftware (IDE) y trabajar de forma autónoma una vez hemos cargado en su memoria el programa que queremos que ejecute. En nuestro caso gobernaremos el Arduino mediante su conexión USB. La configuración de nuestro dispositivo Arduino (como ya fue expuesto en el apartado de software) requiere de la carga en memoria de un programa (LIFA_BASE) que nos proporciona el toolkit “LabVIEW interface for Arduino” para su posterior gobierno por parte del programa desarrollado en LABVIEW.

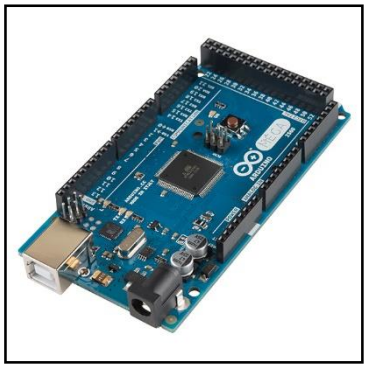


Figura 27. Arduino Mega

A modo de advertencia hay que mencionar que una posible fuente de fallos se encuentra en éste dispositivo debido a que puede no iniciarse correctamente sin motivo aparente (el dispositivo no arranca y no permite que nuestro programa se desarrolle con normalidad). Hemos visto de forma empírica que esto puede ser causa de una mala conexión en los terminales que comunican nuestro Arduino con el resto de elementos del sistema, por lo que habrá que prestar una gran atención en este punto del proceso.

4.3. SISTEMA DE RELÉS

Una vez tenemos acceso eléctrico individual a cada una de las células que componen un sustrato el paso lógico es diseñar un sistema capaz de realizar de forma consecutiva la caracterización de cada una de las células. Esto lo conseguimos mediante el uso de relés electromecánicos. En nuestro caso hemos seleccionado el Módulo Relé 8X para ARDUINO.

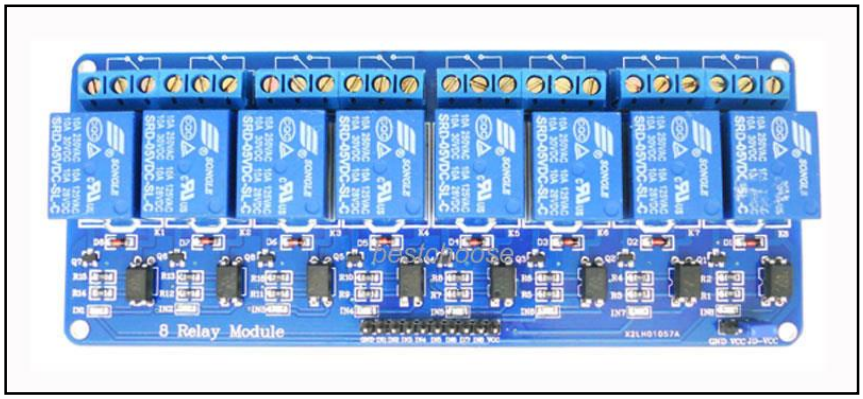


Figura 28. Kit de 8 reles para Arduino

Éste módulo consta de 8 relés capaces de soportar corrientes de 10A a 230V, más que suficiente para cubrir nuestras necesidades. También integran el circuito lógico que permitirá excitarlos directamente mediante una señal digital de 3,3 a 5Vdc. Siguen una lógica negativa, a saber, un relé estará cerrado cuando reciba 0Vdc como señal, y abierto cuando reciba entre 3,3

y 5Vdc; estas consideraciones son fundamentales a la hora de desarrollar el programa que los gobernará.

La conexión de los relés requiere de una fuente de tensión externa (12Vdc) que alimentará sus bobinas y una señal de control de entre 3,3 y 5Vdc proveniente del Arduino (más detalles en Anexo I).

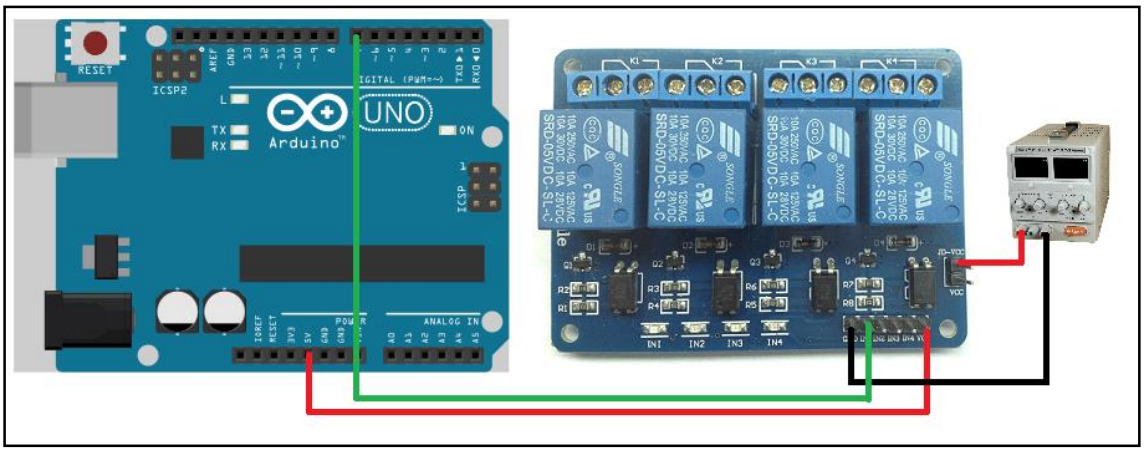


Figura 29. Esquema de conexión de relés para Arduino

4.4. SENSOR DE TEMPERATURA Y HUMEDAD DHT11

El DHT11 es un sensor de temperatura y humedad de salida digital calibrada. Su utilidad será dejar constancia de las condiciones ambientales a las que se realizará cada experiencia.

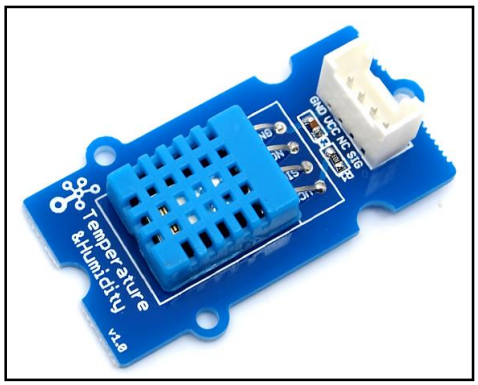


Figura 30. Sensor de temperatura y humedad DHT11

El DHT11 utiliza únicamente un canal de transmisión digital para dar una lectura de estas dos variables (11). El DHT11 está compuesto por un sensor de humedad de tipo resistivo y una resistencia variable NTC para la medida de temperatura; ambos elementos están conectados a

un microcontrolador que codifica y envía la señal digital que posteriormente interpretará Arduino (12). Sus principales características se presentan en la Tabla 2.

Tabla 2. *Características del sensor de temperatura y humedad DHT11*

Rango de medida	Precisión en temperatura	Precisión en humedad
20-90%RH 0-50ºC	±5%RH	±2ªC

Para su puesta en funcionamiento necesitamos únicamente de la alimentación de 5Vdc (VDD) que proporciona Arduino y una resistencia de 4,7KΩ. El sensor se conectará a una entrada digital (en nuestro caso la función que utilizaremos para tomar las lecturas indica que debe ser el pin 2) de nuestro Arduino (DATA) y a la alimentación como se muestra a continuación:

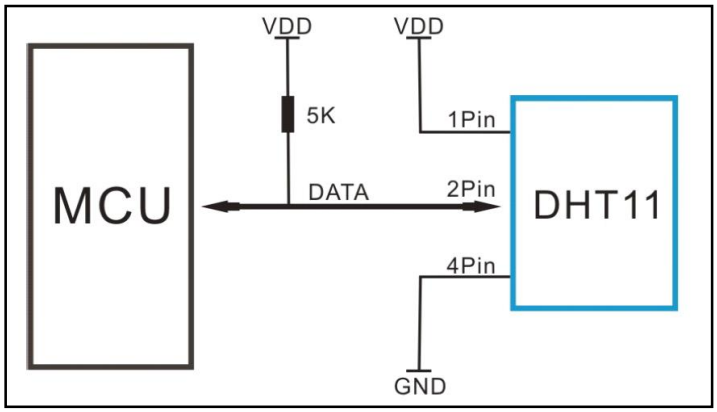


Figura 31. Esquema de conexión para sensor DHT11(13)

Para su correcto funcionamiento requiere de una configuración especial del software. De una forma similar a como instalamos los complementos de Arduino para LABVIEW en este caso tenemos una función previamente desarrollada que nos permite obtener de forma directa nuestras medidas de temperatura y humedad.

Para instalarlas tendremos que cambiar el firmware que introducimos en nuestro Arduino para que se comunique con LABVIEW (LIFA_Base). El nuevo firmware además de la función que nos permitirá interaccionar con el sensor lo podemos obtener del foro de "National Instruments" (https://decibel.ni.com/content/thread/13453?start=30&tstart=0).

La función que utilizaremos en nuestro programa para tomar las lecturas será la siguiente:

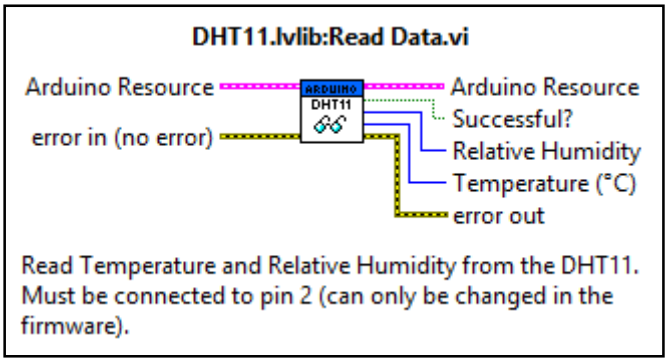


Figura 32. Función para medir temperatura y humedad con el sensor DHT11 en LABVIEW

Como podemos ver tiene 2 valores de entrada que son: la dirección del Arduino que comunicara con el sensor DHT11 y el registro de errores. Las salidas nos ofrecen un booleano que nos indica si la comunicación Arduino-DHT11 ha sido correcta, y otras dos salidas con los valores respectivos de temperatura y humedad.

5. SOLUCIÓN/IMPLEMENTACIÓN SOFTWARE

Una vez vistos los materiales hardware de los que disponemos en nuestra instalación necesitamos crear un software capaz de gestionar todos estos recursos. En nuestro caso vamos a continuar utilizando la plataforma LABVIEW. Partiremos del programa ya visto en los antecedentes, pues sigue siendo perfectamente válido para tomar caracterizar cada célula. A éste habrá que añadirle distintos elementos que sean capaces de gestionar los relés, el sensor de temperatura y la comunicación con Arduino; además implementaremos un sistema cíclico de toma de datos que nos permita realizar ensayos de degradación que pueden llegar a durar más de un día y un submenú capaz de diferenciar las áreas de cada célula.

5.1. INTRODUCCIÓN A LABVIEW

Comenzaremos haciendo una breve introducción a LABVIEW. Explicaremos el funcionamiento de los elementos más básicos, los cuales nos permitirán comprender las diferentes partes de nuestro programa.

Es importante recordar que LABVIEW es una plataforma de programación gráfica, por lo que los distintos elementos están representados por bloques, los cuales tienen entradas y salidas conectadas por cables que conducen la información de unos a otros. No entraremos a describir el funcionamiento más básico, sino las partes de tipo estructural.

5.*1.1. Estructura secuencial*

Este elemento nos permite ejecutar de forma secuencial diferentes partes del programa. Cada marco corresponde con una etapa del proceso que ejecutará de izquierda a derecha. Así podemos forzar a que ciertas tareas tengan que ejecutarse y concluir antes de que inicien otras.

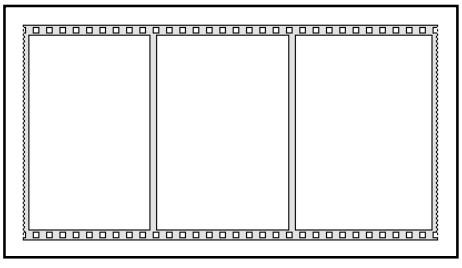


Figura 33. Estructura secuencial

Un caso particular en esta estructura es la “secuencia apilada” en la que los diferentes marcos quedan superpuestos y la forma verlos es mediante un menú en la parte superior del marco.

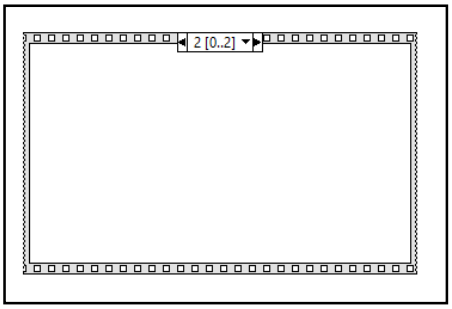


Figura 34. Estructura secuencial apilada

5.1.2. Bucle "for"

Con el bucle for ejecuta el contenido de su interior un número determinado de veces. Este bucle necesita que se le indique el número de veces que queremos que se ejecute.

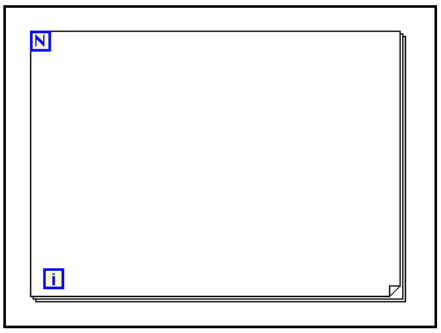


Figura 35. Bucle "for"

Como podemos observar el bucle necesita que le indiquemos el número de repeticiones en "N". A su vez nos permitirá conocer en que ejecución nos encontramos mediante "i".

5.1.3. Estructura "case"

Este tipo de estructuras nos permite ejecutar un código u otro dependiendo de nuestras necesidades. El código que ejecute dependerá del valor de una variable de entrada, esta variable puede ser booleana (true-false) o tener un valor numérico si el número de casos es mayor a dos.

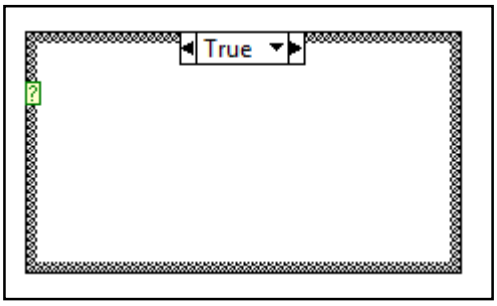


Figura 36. Estructura "Case"

5.1.4. Nodo de fórmulas

Es un elemento que permite desarrollar fórmulas matemáticas complejas (que como veremos son difíciles de implementar en el lenguaje propio de LABVIEW) así como utilizar los elementos de programación antes descritos (bucles “for”, condicionales “if-else”, etc.) de la forma tradicional.

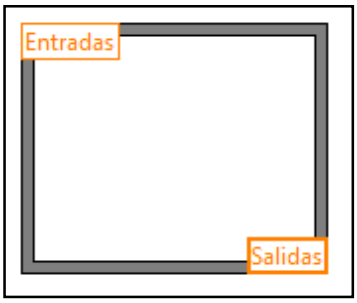


Figura 37. Nodo de fórmulas

Como podemos ver consta de un cajón en el que escribir (texto plano) nuestra expresión. A la izquierda encontramos un ejemplo de cómo definir una entrada para pasarle datos y a la derecha una salida que nos comunicará los resultados.

5.1.5. Operaciones básicas

LABVIEW permite realizar operaciones de todo tipo de entre las que destacamos las más básicas como son la suma, resta, etc. Estas operaciones tienen diferentes entradas y salidas dependiendo del tipo, a modo de ejemplo veremos la conexión de una suma básica.

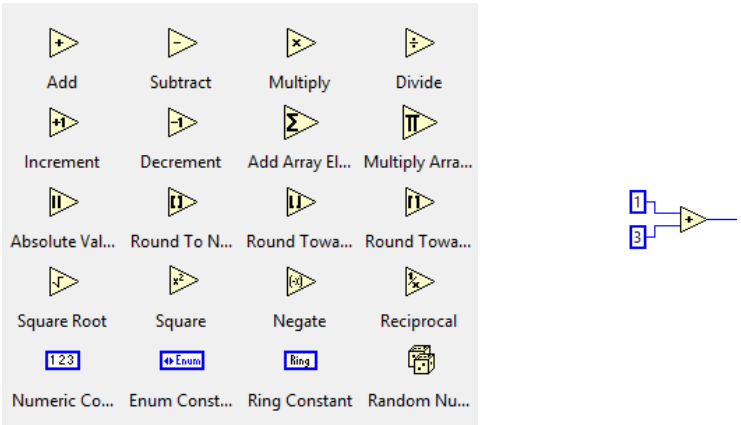


Figura 38. Paleta de operaciones básicas y ejemplo

5.1.6. Arrays

En nuestro caso serán de gran utilidad las matrices o “arrays” como medio para guardar y manipular los datos.

Pasamos a describir de forma muy rápida las funciones más utilizadas.

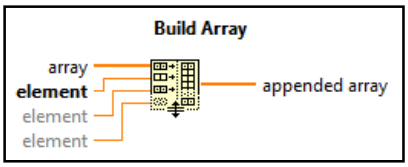


Figura 39. Función "Build Array" en LABVIEW

Con "Build Array" podemos crear un array a partir de datos sueltos u otro array.

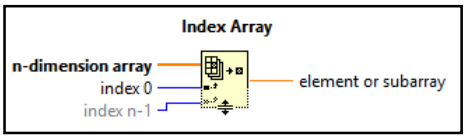


Figura 40. Función "Index Array" en LABVIEW

"Index Array" nos permite acceder a los elementos individuales que componen el array.

5.1.7. Comunicación GPIB

Con estas funciones podremos comunicarnos con los elementos que estén conectados mediante GPIB. Solo tendremos que indicarle la dirección de nuestro aparato y los datos que queremos transmitirle.

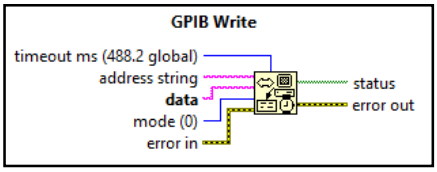
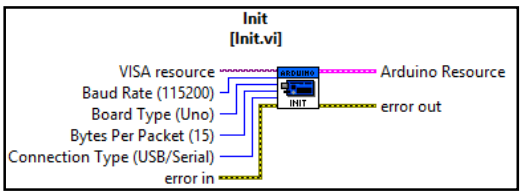


Figura 41. Función "GPIB Write" en LABVIEW

5.1.8. Arduino

Hay gran número de funciones para comunicarse con Arduino.

Las primeras funciones son las que necesitamos para iniciar y finalizar la comunicación con nuestro Arduino. Estas funciones necesitan recibir en todo caso una cadena de texto que le indique con qué Arduino nos estamos comunicando; también pueden unirse con un seguimiento de error.



Figuras 42. Función "Init" (iniciar Arduino) en LABVIEW

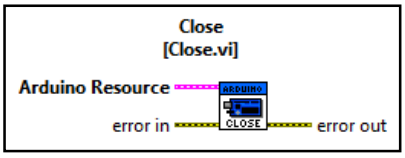


Figura 43. Función "Close" para Arduino en LABVEIW

Otras funciones importantes son las que utilizaremos para modificar el estado de los pins digitales del Arduino. Así primero tendremos que indicarle a Arduino que queremos utilizar cierto pin como salida o entrada digital, y si seleccionamos la opción salida hay que indicar el valor de salida, que puede ser low (0) o high (1).

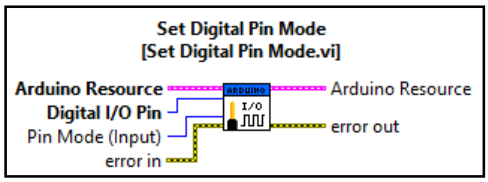


Figura 44. Función "Set Digital Pin Mode" para Arduino en LABVIEW

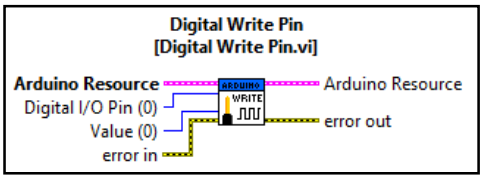


Figura 45. Función "Digital Write Pin" para Arduino en LABVIEW

5.1.9. Funciones

También podemos encontrar funciones que sean definidas por otros usuarios. A continuación mostraremos algunos ejemplos.

Una función que utilizaremos será la que controle nuestro sensor de temperatura y humedad "DHT11". Esta función fue desarrollada por otro usuario de LABVIEW, lo que nos permite descargarla y utilizarla directamente. Esta función como el resto de elementos tiene una serie de entradas, realiza una tarea, y posteriormente devuelve uno o varios valores de salida.

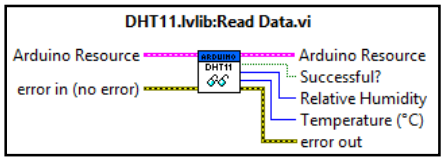


Figura 46. Función para lectura de temperatura y humedad con DHT11 para Arduino en LABVIEW

Si abrimos esta función podemos ver el código que se ejecuta cuando la llamamos.

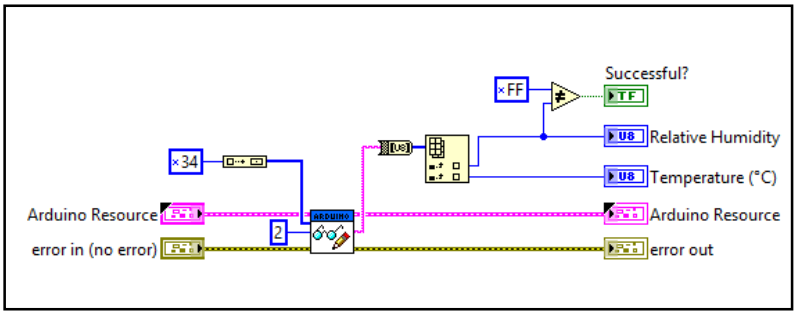


Figura 47. Diagrama de bloques de la función “DHT11.lvLibRead Data”

6. DESCRIPCIÓN DEL PROGRAMA

Como ya hemos dicho necesitamos de una rutina o programa capaz de realizar las nuevas actividades que nos permite el hardware descrito con anterioridad; en última instancia pretendemos hacer el proceso de caracterización lo más automático posible, reduciendo las tareas a realizar a la colocación de los sustratos y posterior interpretación de los resultados.

Así nuestro programa debe ser capaz de, interactuando con el hardware disponible, caracterizar hasta 4 sustratos (y una célula adicional utilizando un sistema de pinzas de cocodrilo, similar al que había disponible con anterioridad) de forma consecutiva para exportar los datos obtenidos en un fichero (en nuestro caso un .txt) que permita su posterior interpretación.

Nuestro programa partirá del programa ya descrito en los antecedentes. Si recordamos, este programa, permitía obtener una curva característica para un sustrato mediante la aplicación de una serie de potenciales a una célula mediante una fuente controlada de tensión mientras simultáneamente toma unas lecturas de corriente.

Esta operación es recurrente: hay que realizarla cuatro veces en cada sustrato, además contamos con la posibilidad de trabajar con hasta 4 sustratos en el mismo ensayo(lo que nos permitirá caracterizar hasta 16 células), y añadimos la posibilidad de realizar ensayos de degradación (dependiendo de las necesidades pueden llegar a realizarse varios ensayos por hora durante varios dias).

6.1. INTERFAZ

Comenzaremos la descripción por su interfaz gráfico. En nuestro caso partimos del interface original, el cual ha sido modificado para acoger las modificaciones que hemos añadido. Así contamos con una pantalla principal bastante intuitiva que nos permitirá seguir nuestro experimento.

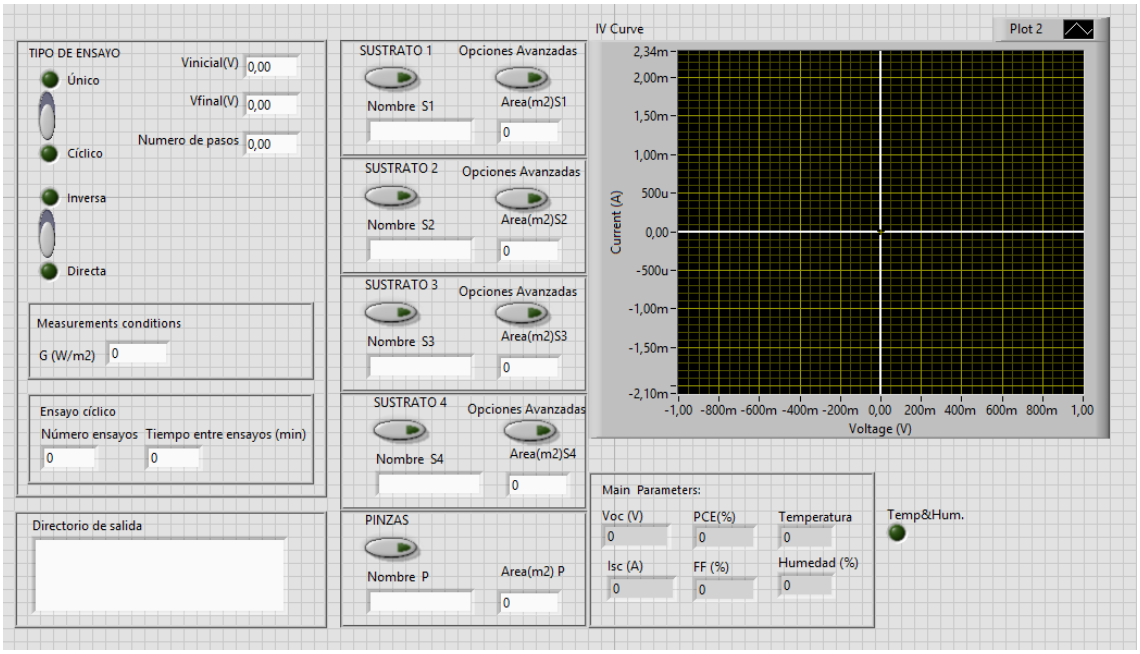


Figura 48. Panel frontal o interface del programa desarrollado

Como podemos ver mantiene una serie de elementos básicos:

- Entrada de los parámetros con los que realizar la caracterización: Tensión inicial (Vinicial), Tensión final (Vfinal), Número de pasos, Irradiancia (G).
- Datos obtenidos en el ensayo: Gráfico I-V (Plot2-IV Curve), tensión de circuito abierto (Voc), corriente de cortocircuito (Isc), tensión de máxima potencia (Vmpp), corriente de máxima potencia (Impp), eficiencia (PCE), factor de llenado (FF).

A estos elementos hemos tenido que añadir un grupo de nuevos parámetros.

Los únicos elementos de salida que se han añadido son los indicadores de Temperatura (en ºC) y la Humedad relativa (en %), los cuales mostraran los valores obtenidos durante un ensayo una vez concluido este. Además hemos añadido la representación de los ejes de coordenadas en la representación gráfica para poder ubicar la curva de forma más intuitiva.

Tenemos un nuevo sistema de almacenamiento de archivos debido a que nos encontramos con dos problemas en la versión anterior: al existir la posibilidad de ensayar con varios sustratos necesitamos nombrarlos de forma independiente; además contamos con la posibilidad de realizar ensayos muy largos en los que no podemos esperar al final para almacenar los archivos con los datos obtenidos, como hacia la versión anterior, así hemos añadidos la posibilidad de indicar el directorio en el que almacenar los archivos de datos obtenidos en cada ensayo. La opción que hemos elegido es copiar la dirección de la carpeta en que queremos guardar los datos y pegarla en la pantalla principal.

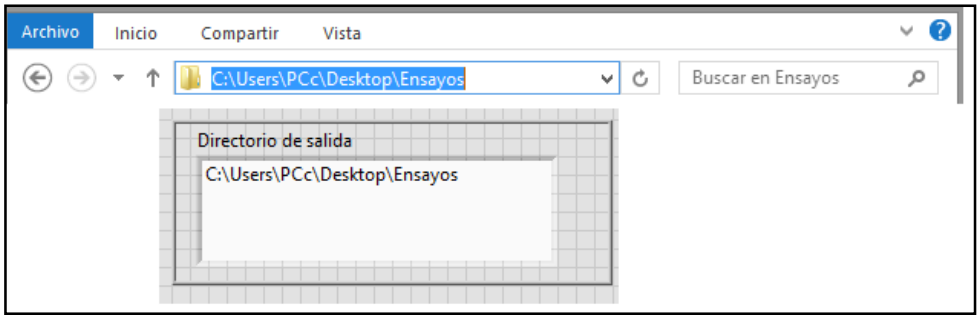


Figura 49. Ejemplo de asignación de directorio

Hemos añadido la opción de elegir si el tipo de célula con la que vamos a ensayar es de arquitectura estándar o "directa", en la que el ITO o parte común está conectada al terminal positivo de la fuente de tensión, o arquitectura "inversa", en la que la polaridad se invierte.

Otra nueva opción es la que indica el tipo de ensayo a realizar. Los ensayos podrán ser "únicos" si solo va a trazarse una curva I-V o "cíclicos" en caso de realizar ensayos de larga duración como son los de degradación.

En el caso de que seleccionemos hacer un ensayo cíclico tendremos que indicar la cantidad de ensayos que queremos realizar así como el tiempo que queremos dejar pasar entre 2 ensayos consecutivos.

Es importante en este punto dejar indicado que el tiempo debe ser mayor que el necesario para caracterizar los sustratos, de no ser así realizará los ensayos de forma consecutiva.

No siempre que realicemos un experimento contaremos con un múltiplo de 4 sustratos por lo que puede convenirnos seleccionar que número de sustratos queremos ensayar. Por eso hemos habilitado un pequeño menú individual para cada sustrato a caracterizar.

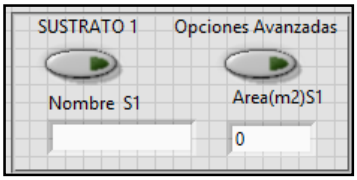


Figura 50. Menú de opciones para un sustrato

Los campos con los que contamos son 2: el área del sustrato y su nombre (será el nombre que tome el fichero que guarde los datos del experimento). En la parte superior izquierda tenemos el botón que permite seleccionar un sustrato para el experimento: cuando se inicie el experimento, en caso de ser pulsado (el pulsador se tornará verde), se realizará una caracterización de dicho sustrato.

Otra opción que hemos añadido es la de "Opciones Avanzadas". Esta opción desplegará un menú cuando se inicie el programa.

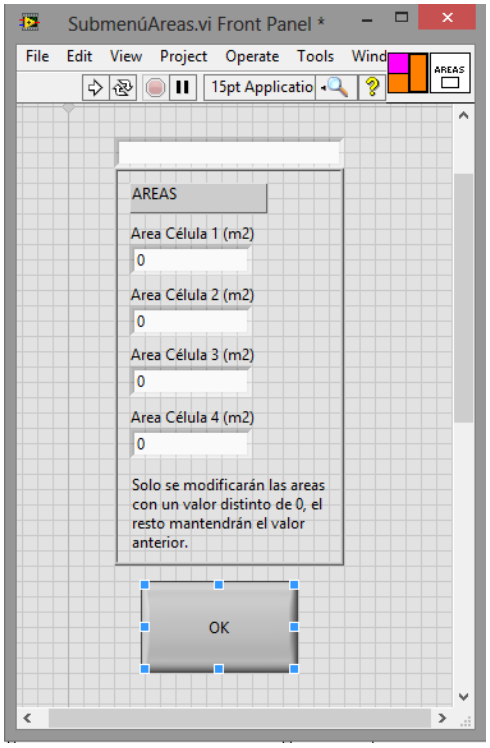


Figura 51. Menú de "Opciones avanzadas"

Esta pantalla emergente aparecerá en caso de haber seleccionado "Opciones Avanzadas". En la parte superior nos mostrará el sustrato al que hace referencia y nos permitirá modificar las áreas de las distintas células que componen el sustrato en caso de encontrar diferencias significativas. Si una casilla mantiene el valor "0" no modificará el área de célula indicada en la pantalla principal. Una vez rellenados los campos pulsaremos el botón "OK" permitiendo al programa continuar.

6.2. CODIGO DEL PROGRAMA

Vamos a presentar el programa desarrollado en 6 etapas (dispuestas en una estructura secuencial de 6 bloques), en cada una de las cuales podrán desarrollarse varias tareas de forma simultánea. Como hemos indicado con anterioridad, el programa se encargará de la gestión del hardware que hemos instalado, es por eso que la mayoría de las tareas que ejecuta se encargan de configurarlo para poder realizar una caracterización.

El proceso de caracterización esta tomado del programa original; como veremos ha sido ligeramente modificado para salvar varios problemas que han surgido, así como convertido en una subrutina de nuestro programa principal. Es por este motivo por el que lo hemos convertido en una función independiente a la cual llamaremos cuando sea necesario de caracterizar una célula. Podemos presentar un esquema general de nuestro programa a través del siguiente flujograma.

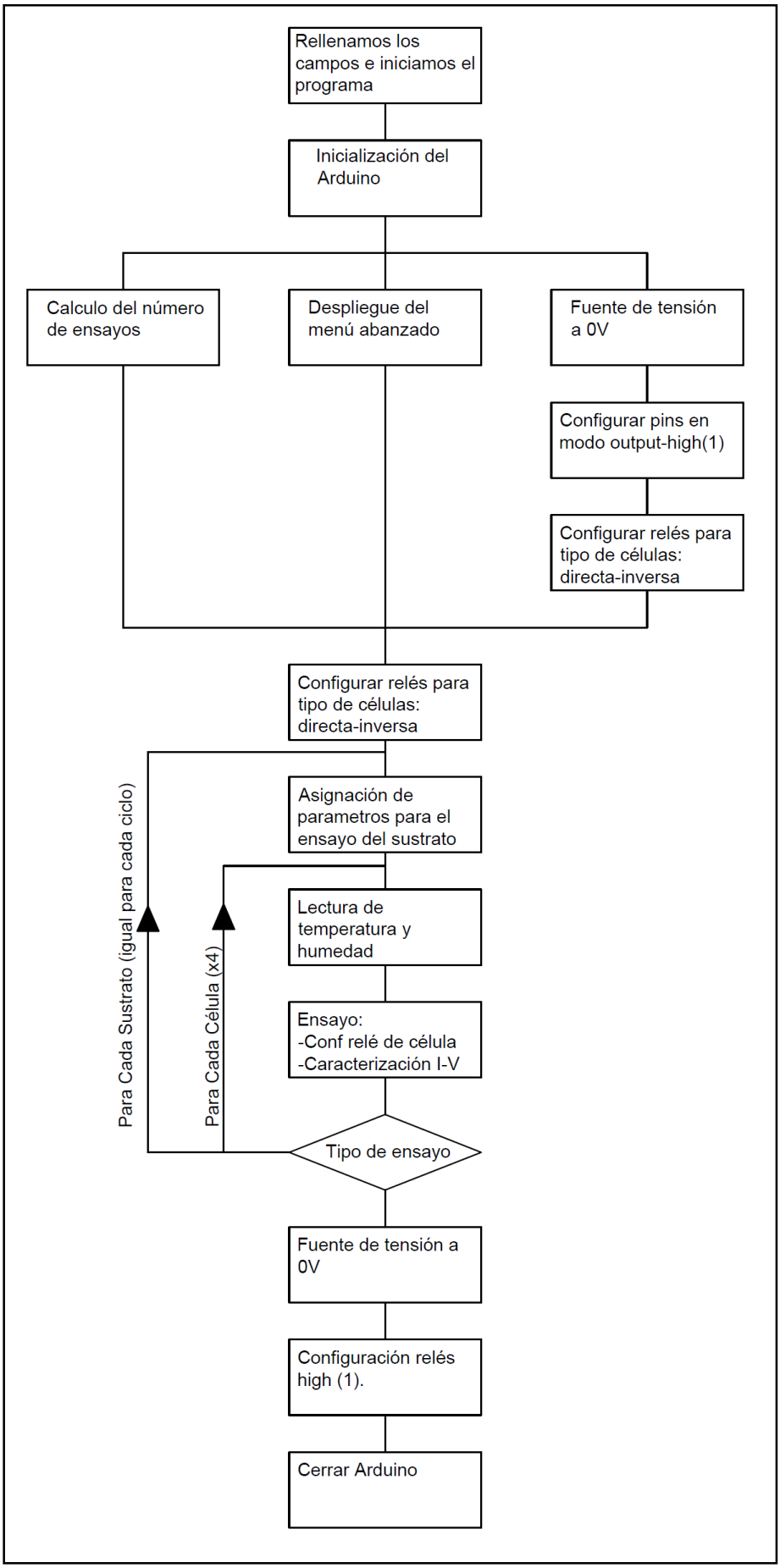


Figura 52. Flujograma del programa desarrollado

6.2.1. Inicio Ardiuno

El primer paso que realiza nuestro programa es iniciar el dispositivo Arduino, permitiendo así su comunicación con LABVIEW.

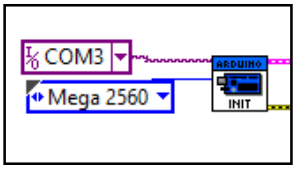


Figura 53. Inicialización de Arduino en el programa desarrollado

Este apartado es fundamental debido a que nos provee de un "localizador" para nuestro Arduino. Este localizador dirá al resto de funciones donde puede encontrar al Arduino mediante la conexión del "cable" rosa ("Arduino Resource").

6.2.2. Preparación de los ensayos

En esta etapa se desarrollan distintas tareas de forma simultánea. En una de ellas se cargan prácticamente todas la variables que van a regir nuestro ensayo, en otra se inicializarán los relés, quedando dispuestos para su utilización en etapas posteriores.

a) Tipo de ensayo

Este bloque definirá el número de iteraciones que llevará a cabo cada bucle posterior en función del tipo de ensayo.

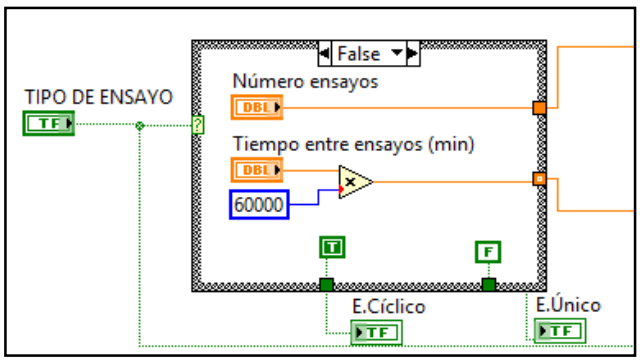


Figura 54. Calculo de parámetros del ensayo en el programa desarrollado

Esta estructura "case" definirá el tipo de ensayo. Si queremos realizar un ensayo cíclico hará que la prueba se repita "n" veces o una solo en caso de querer un ensayo único. También calculará el tiempo entre ensayos.

b) Áreas y menú de opciones avanzadas

Este menú introduce en nuestro programa el valor de las áreas de las distintas células. Si el usuario así lo indicase será el encargado de llamar a la función "Opciones Avanzadas", la cual nos permite modificar el área de las distintas células de un sustrato.

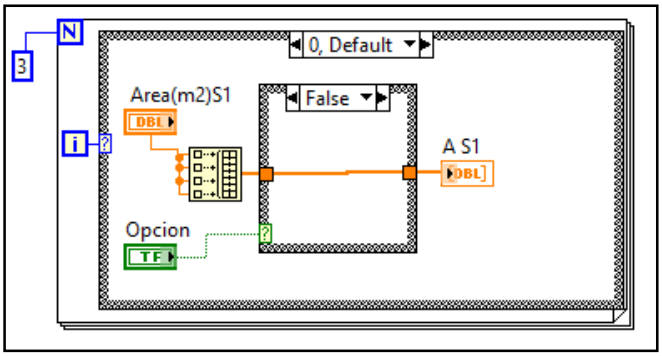


Figura 55. Asignación de áreas en el programa desarrollado

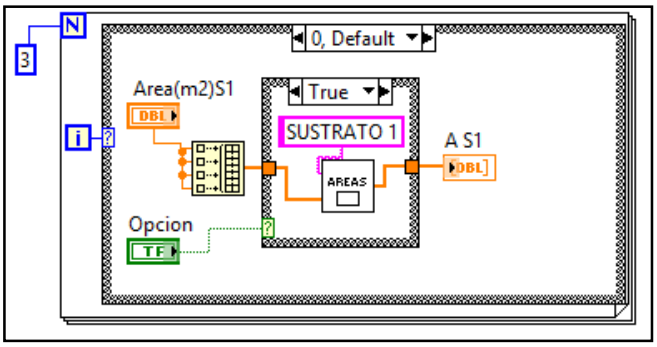


Figura 56. Asignación de áreas para "Opciones Avanzadas" en el programa desarrollado

Esta estructura, integrada en un bucle "for", se repite para cada sustrato. En caso de no haber seleccionado "Opciones Avanzadas" asignará la misma área a las 4 células guardando dichos valores en un vector (primera componente para primera célula, segunda componente para la segunda células, etc.). En caso de seleccionar "Opciones Avanzadas" ejecutará la función "AREAS". Esta función desplegará el menú que ya hemos comentado y trabajará con el siguiente código:

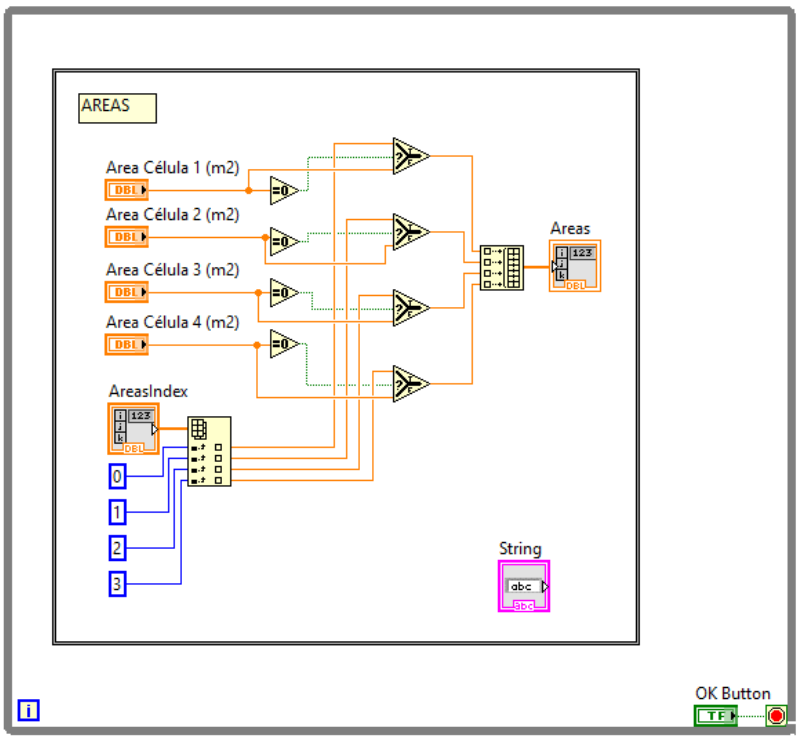


Figura 57. Diagrama de bloques del menú de "Opciones Avanzadas"

Dentro de un bucle "while" (que mantendrá el menú activo mientras no pulsemos el botón "OK") asignamos a cada valor del array de áreas el valor que indiquemos siempre y cuando sea distinto de "0", caso en el que conserva el valor indicado en la pantalla principal.

c) Inicializar relés

Quizás la parte más compleja de esta etapa sea la encargada de inicializar los relés.

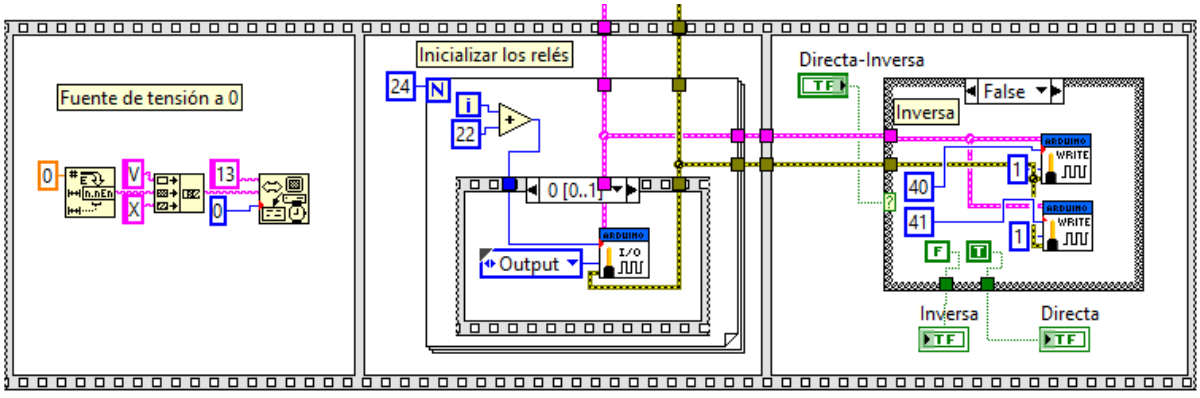


Figura 58. Etapa de inicialización de relés

Podemos apreciar que se trata de otra secuencia. Primeramente pondremos la fuente de tensión a 0Vdc para evitar provocar un cortocircuito al colocar los terminales de esta fuente en su posición de ensayo "directa" o "inversa".

En la segunda parte configuramos todos los pins correspondientes a los relés como salidas digitales y posteriormente le asignamos el valor “high” (un “1” lógico, es decir 5Vdc). La razón de poner nuestros pins en “high” es que la lógica de nuestros relés es negativa, por lo que cuando reciban un “1” lógico mantendrán la posición que tienen cuando no excitamos su bobina. Esta operación se repetirá una vez por cada relé (24 veces) asegurando que todos los relés se encuentren en su posición de reposo al iniciar el programa.

La última tarea que realizamos es preparar los contactos de la fuente de tensión mediante 2 relés. Dependiendo de lo que hayamos seleccionado en el panel principal activará los relés 40 y 41 o los desactivará permitiendo una conexión u otra.

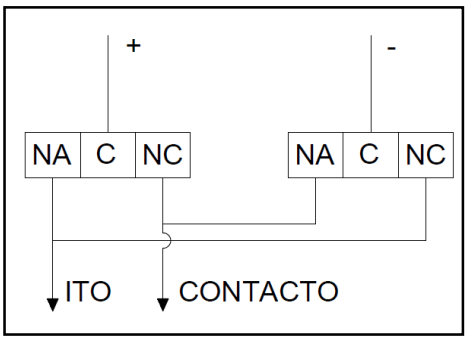


Figura 59. Esquema de conexiones de los relés encargados de la conexión directa-inversa

Como podemos apreciar en la imagen la opción “directa” conectará el ITO al polo positivo y el contacto de aluminio al polo negativo (los dos relés estarán sin excitar) y la opción “inversa” conectará el ITO al polo negativo de la fuente y el contacto de aluminio al polo positivo

6.2.3. Caracterización

En la etapa de caracterización es la más compleja de todas debido a la cantidad de bucles que en ella encontramos. Estos bucles se encargarán de repetir cada etapa del proceso en función del tipo de ensayo que demandemos. Por ejemplo: para un ensayo en el que tengamos dos sustratos y queramos hacer una prueba de degradación (un ensayo por hora durante 24h) necesitaremos caracterizar 2 sustratos distintos, compuestos por 4 células cada uno; a su vez todo esto se realiza 24 veces en 24 horas.

Este proceso lo podemos ilustrar mediante un flujograma:

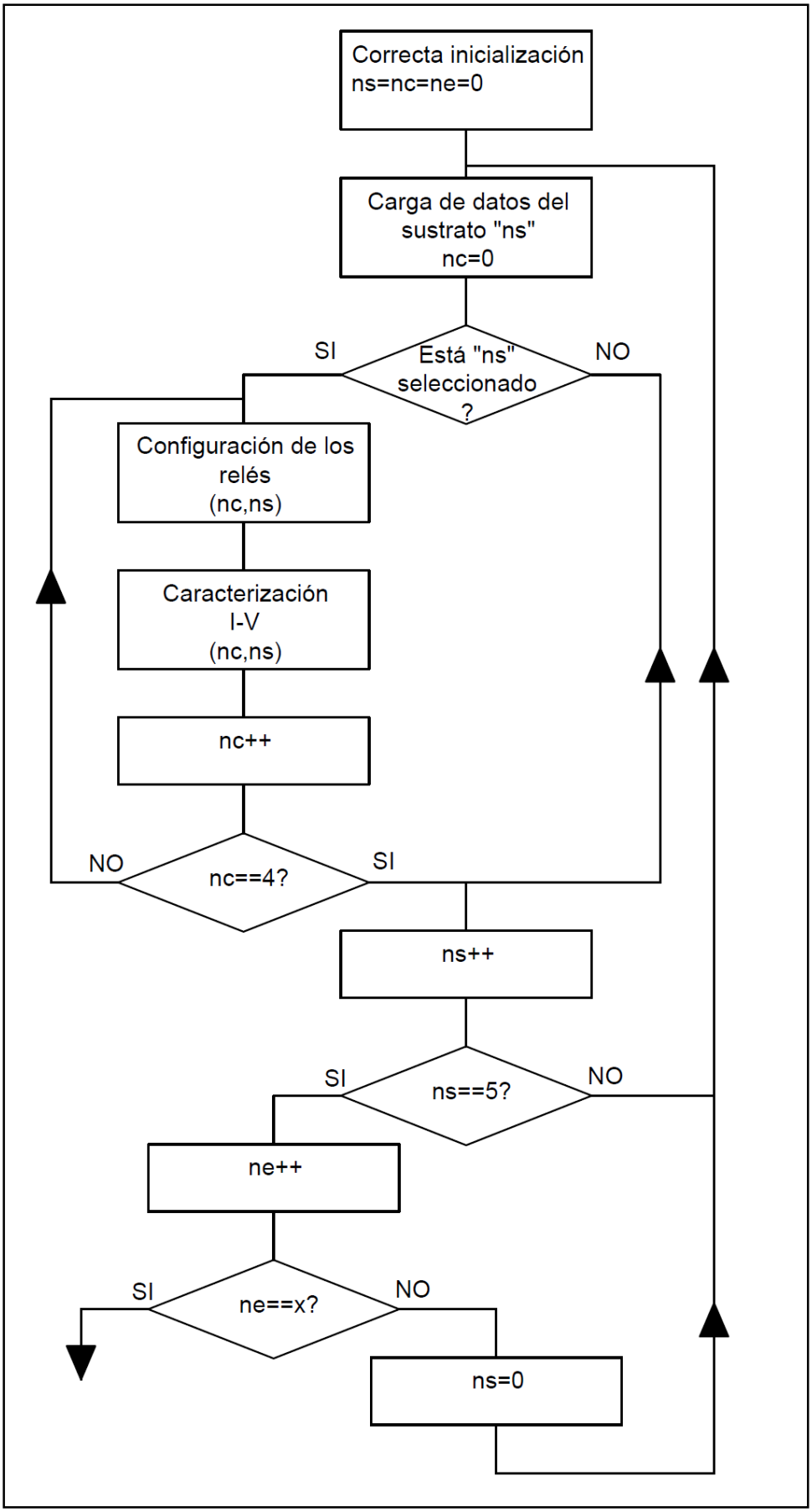


Figura 60. Flujograma del proceso de caracterización

Como podemos observar cuando se inicia el proceso comprobamos qué sustratos han sido seleccionados para la caracterización.

Si un sustrato ha sido seleccionado pasará a caracterizarse de forma consecutiva cada una de las células que lo componen (4 veces); para esto se le pondrá en contacto con la fuente de tensión mediante el sistema de relés para posteriormente trazar la curva I-V.

Una vez terminado un sustrato se comprueba si hay que caracterizar el siguiente; en caso afirmativo se repite el proceso anterior, y en caso negativo se pasa al siguiente sustrato.

Cuando todos los sustratos han sido comprobados y/o caracterizados se comprueba si hay que realizar más de un ensayo, de ser así se vuelve al principio tantas veces como ensayos tenga nuestra prueba.

Visto en un plano más general el programa queda de la siguiente forma:

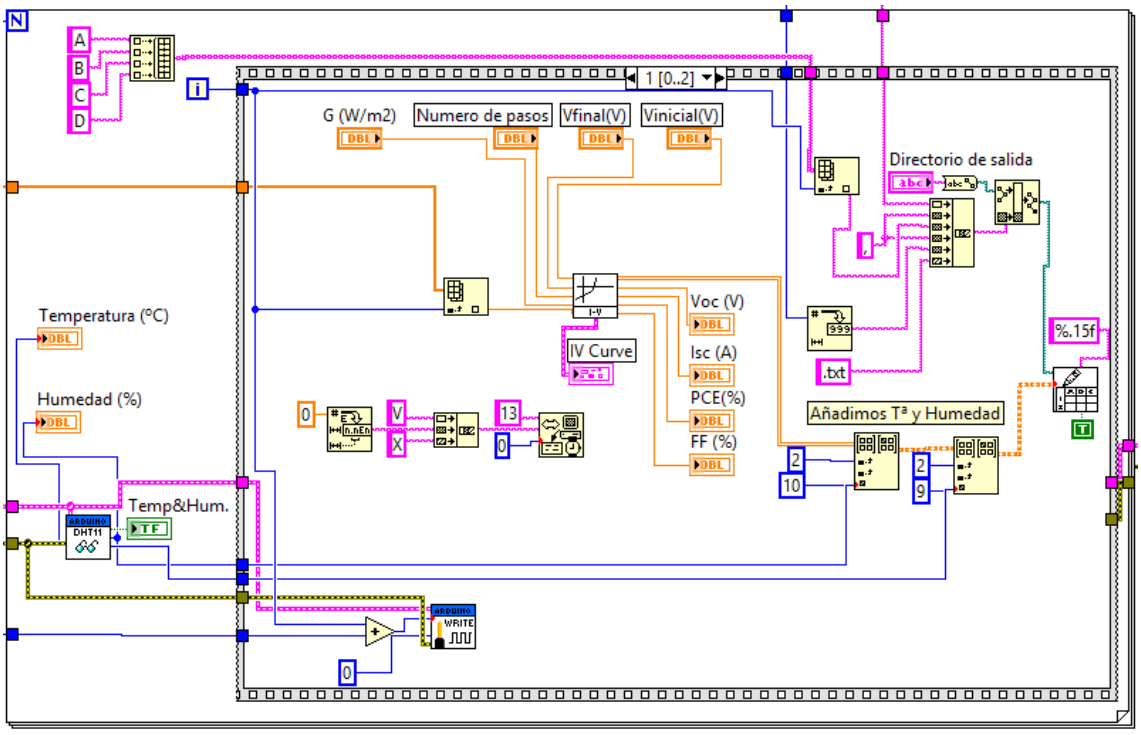


Figura 61. Diagrama de bloques del proceso de caracterización

Este código representaría la parte de nuestro programa que caracteriza una célula, repitiéndose 4 veces por sustrato. Como podemos observar antes de caracterizar una célula se llama a la función “DHT11” (vista anteriormente) que se encargará de registrar las condiciones ambientales en las que se ha desarrollado el ensayo.

El paso siguiente (Diagrama estructurado) excitará el relé que nos dé acceso a nuestra célula para después llamar a la función “IV Curve”. Por último creará un fichero de texto (.txt) en el que quedará constancia de los datos obtenidos durante nuestro ensayo.

6.2.4. Función I-V

La nueva función encargada de trazar la curva I-V se llama “IVconPCEyFFsinMathScript.vi”. Esta función variara los valores de la fuente de tensión registrando las variaciones de corriente, obteniendo así la curva I-V de nuestra célula.

El código de esta función es prácticamente igual al código del programa de partida. Su función es hallar la curva I-V de una sola célula.

Sus variables de entrada son las condiciones del ensayo:

- Tensión inicial.
- Tensión final.
- Número de pasos.
- Irradiancia.
- Área de la célula.

Y sus salidas:

- Matriz de resultados.
- Otros datos ya contenidos en el punto anterior como son la tensión de circuito abierto, corriente de cortocircuito, tensión y corriente de máxima potencia, PCE y FF.
- Datos para representación gráfica.

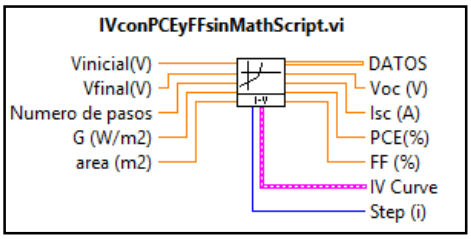


Figura 62. Función “I-V” en LABVIEW

Mostraremos el código de esta función en dos partes bien diferenciadas. La primera parte de la función se muestra en la figura 63.

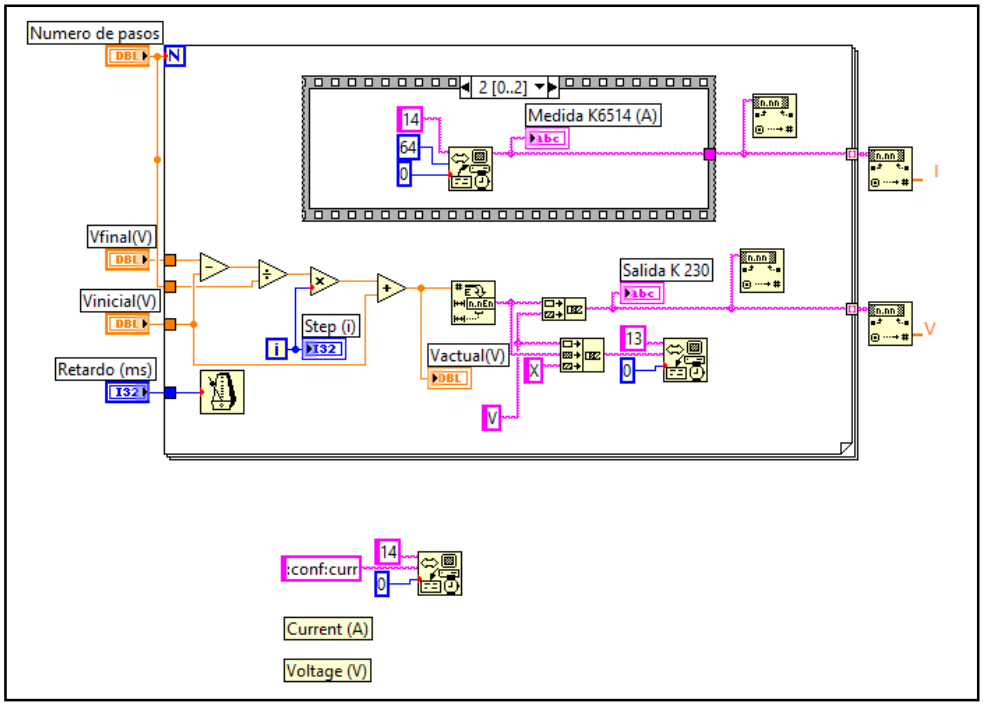


Figura 63. Diagrama de bloques de la función I-V (I)

Podemos apreciar que esta primera parte de la función es la que se encarga de recopilar los datos mediante el control de la fuente de tensión y el electrómetro como ya vimos en los antecedentes

Una segunda parte de este código se presenta de la siguiente manera en la figura 64.

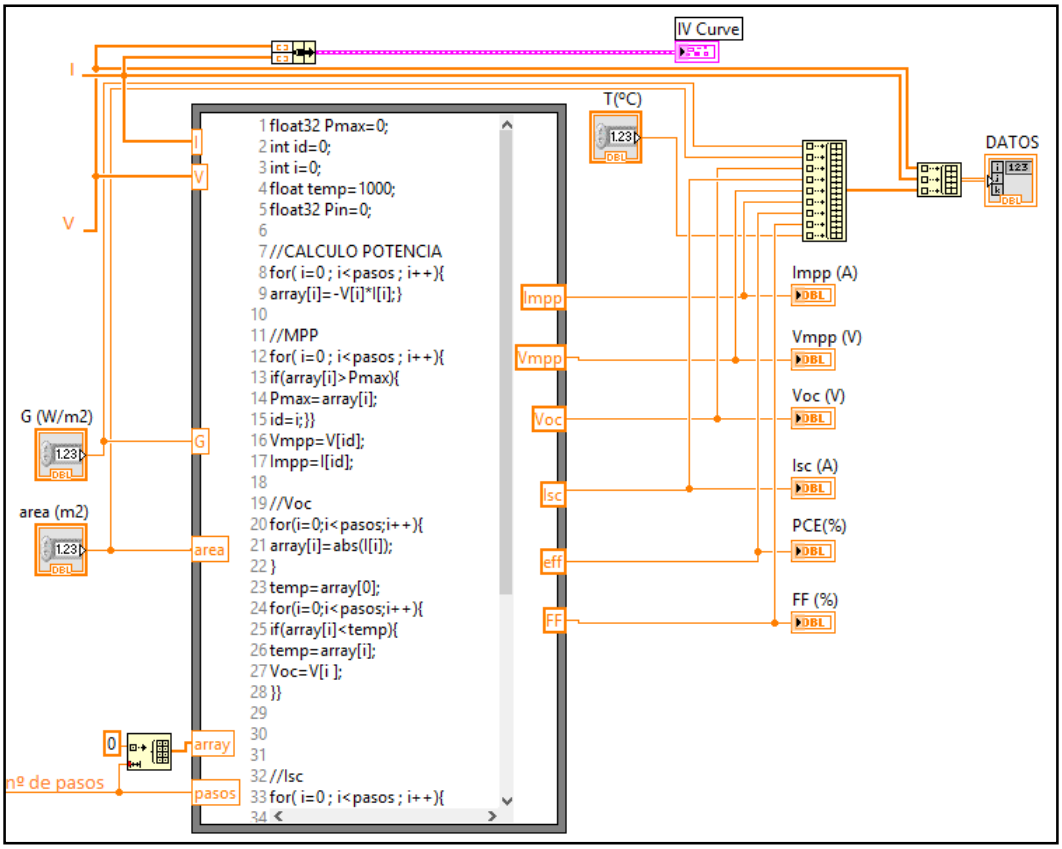


Figura 64. Diagrama de bloques de la función I-V (II)

Esta última parte del código ha tenido que ser modificada debido a que en nuestra versión actual de LABVIEW no disponemos del complemento MATHSCRIPT, el cual nos permitía generar complejas funciones matemáticas. Por eso hemos tenido que sustituir este módulo por un "nodo de fórmulas". Esto nos ha forzado a generar un nuevo código capaz de calcular los principales parámetros de la célula utilizando estructuras básicas de programación como bucles for. Estos parámetros son:

- Punto de máxima potencia: Es fundamental para conocer la tensión de máxima potencia (Vmpp) y la corriente de máxima potencia (Impp). Para hallar el punto de máxima potencia hemos creado una variable auxiliar llamada array del mismo tamaño que la muestra. Esta variable contendrá los valores de potencia calculada como "P=–V*I" en cada punto. Una vez hecho esto podemos localizar el índice del punto de máxima potencia en este array para obtener nuestros valores de "Vmpp" e "Impp". El código es el siguiente:

```
//CALCULO POTENCIA
for( i=0 ; i<pasos ; i++){
array[i]=-V[i]*I[i];}

//MPP
for( i=0 ; i<pasos ; i++){
if(array[i]>Pmax){
Pmax=array[i];
id=i;}}
Vmpp=V[id];
Impp=I[id];
```

- Puntos de corte con los ejes: estos puntos corresponden con los valores de la tensión de circuito abierto (Voc) y la corriente de cortocircuito (Isc). Los dos puntos se hallan de una forma similar: asignamos a array el valor de nuestra magnitud para después localizar en él el mínimo (punto más próximo a 0). Estos valores serán más exactos cuanto mayor sea el número de puntos que tengamos debido a que será más probable que alguno coincida con el cero real. El código encargado de esta operación es:

```
//Voc
for(i=0;i<pasos;i++){
array[i]=abs(I[i]);}
temp=array[0];
for(i=0;i<pasos;i++){
if(array[i]<temp){
temp=array[i];
Voc=V[i ];}}

//Isc
for( i=0 ; i<pasos ; i++){
array[i]=abs(V[i]);}
temp=array[0];
for( i=0 ; i<pasos ; i++){
if(array[i]<temp){
Isc=I[i];
temp=array[i];}}
```

- A continuación calculamos la eficiencia de la célula así como el factor de forma. Este paso simplemente requiere de la implementación de sus formulaciones matemáticas en el código. Las formulas y el código quedan como sigue:

$$FF = \frac{P_{máx}}{I_{sc}V_{oc}} = \frac{(I * V)_{máx}}{I_{sc}V_{oc}}$$

$$\eta = \frac{P_{máx}}{P_{light}} = \frac{I_{sc} * V_{oc} * FF}{P_{light}}$$

```
    //fill factor (%)
    FF=Impp*Vmpp/(Isc*Voc)
*100;

    //eficiencia (%)
    Pin=G*area;
    eff=(Pmax/Pin)*100;
```

Una vez realizada la caracterización de todas y cada una de las células de nuestro ensayo pasamos a la siguiente etapa de nuestro programa.

6.2.5. Relés a punto seguro

La penúltima etapa se encargará de dejar el sistema en el estado inicial. Su cometido será llevar a reposo a los relés que comunicaban la fuente de tensión con las células. Este proceso se realiza en dos sub-etapas:

La primera pone nuestra fuente de tensión a 0Vdc evitando un posible cortocircuito durante la maniobra posterior.

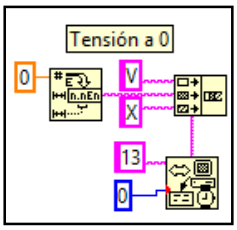


Figura 65. Puesta de tensión a 0Vdc

La segunda lleva los dos relés a su estado de reposo.

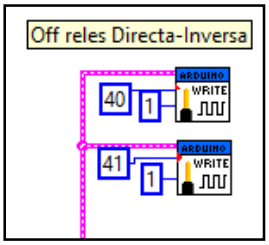


Figura 66. Desactivación de relés al terminar las caracterizaciones

6.2.6. Cierre Arduino

Una vez concluido todo el proceso simplemente falta cortar la comunicación con Arduino para tenerlo disponible en ensayos posteriores.

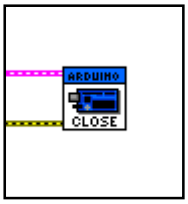


Figura 67. Cierre de Arduino

7. CONCLUSIONES

Una vez vistas todas las partes de que consta el trabajo de forma pormenorizada solo queda dejar patente su materialización.

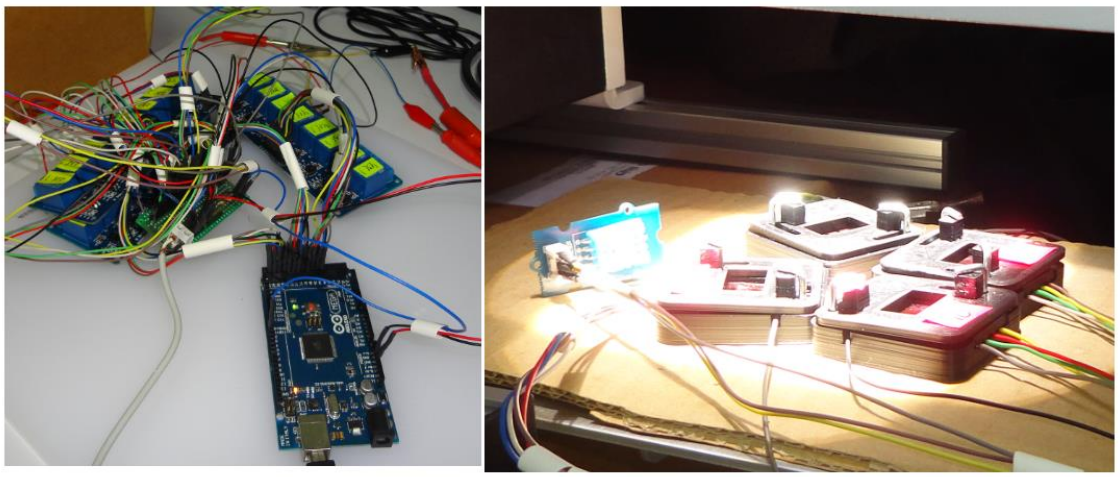


Figura 68. Implementación del proyecto

El sistema, debidamente implementado y puesto a prueba, cumple las metas que nos habíamos propuesto: desarrollar un método automático, cómodo y fiable para caracterizar células solares. Como las células suelen fabricarse en grupos, permite que una vez fabricadas puedan ser puestas a prueba de forma rápida, simultánea y sin dañarlas.

Se ha conseguido reducir al mínimo los problemas relacionados con el deterioro de los contactos, mejorando no solo la consecución de un ensayo sino su reproducibilidad. Con esto garantizamos la correcta conservación de los sustratos que contengan células con buenos índices de funcionamiento.

Gracias a la implementación de este sistema se podrá realizar la caracterización de una tanda de hasta 16 células (4 sustratos) sin la necesidad de perder tiempo cambiando de una célula a otra, ni calidad en la medida debida a un mal contacto. También permitirá seguir utilizando las pinzas cocodrilo para células hechas a partir de sustratos diferentes, así como para ensayos en cámaras de vacío, etc.

7.1. PROPUESTAS DE MEJORA

El alcance del trabajo ha sido el que tanto tiempo como medios disponibles han permitido. Por eso podemos decir que el trabajo tiene aún muchas posibilidades de mejora.

De entre éstas posibilidades podemos destacar el añadir la posibilidad de controlar la temperatura de las células durante los ensayos. Éste proceso, normalmente de calentamiento, se podría llevar a cabo mediante un dispositivo Peltier que, mediante algún material conductor (cobre, aluminio, etc.) modificase la temperatura de las células.

Otra posible mejora es la monitorización de la temperatura en la propia célula durante los ensayos. Este problema (parcialmente abordado) podría ser solucionado integrando un termopar en contacto con las células. Como método de lectura proponemos Arduino. El problema reside en adaptar las señales del termopar a las que Arduino es capaz de leer.

8. BIBLIOGRAFÍA

(1) Nanomorphology – Efficiency Relationship in Organic Bulk Heterojunction Plastic Solar Cells, Dipl. Phys. Harald Hoppe, pag 1

(2) Solar Energy Materials & Solar Cells 92 (2008) 371–373

(3) Matthew O.Reese el al, Consensus stability testing protocols for organic photovoltaic materials and devices 95 (2011) 1254-2156

(4) Dipl.Ing. Klaus Petritsch, Dipl.Ing. Klaus Petritsch, Organic Solar Cell Architectures, (2000) 18-22

(5) Dipl. Phys. Harald Hoppe, Nanomorphology – Efficiency Relationship in Organic Bulk Heterojunction Plastic Solar Cells (2004) 12-14

(6) http://www.ni.com/labview/esa/

(7) KEITHLEY, Model 230 Programable Voltage Source, Instruction Manual

(8) KEITHLEY, Model 6514 System Electrometer, Instruction Manual

(9) http://printeddreams.es/producto/abs-k4-negro/

(10) https://www.arduino.cc/en/Main/ArduinoBoardMega2560

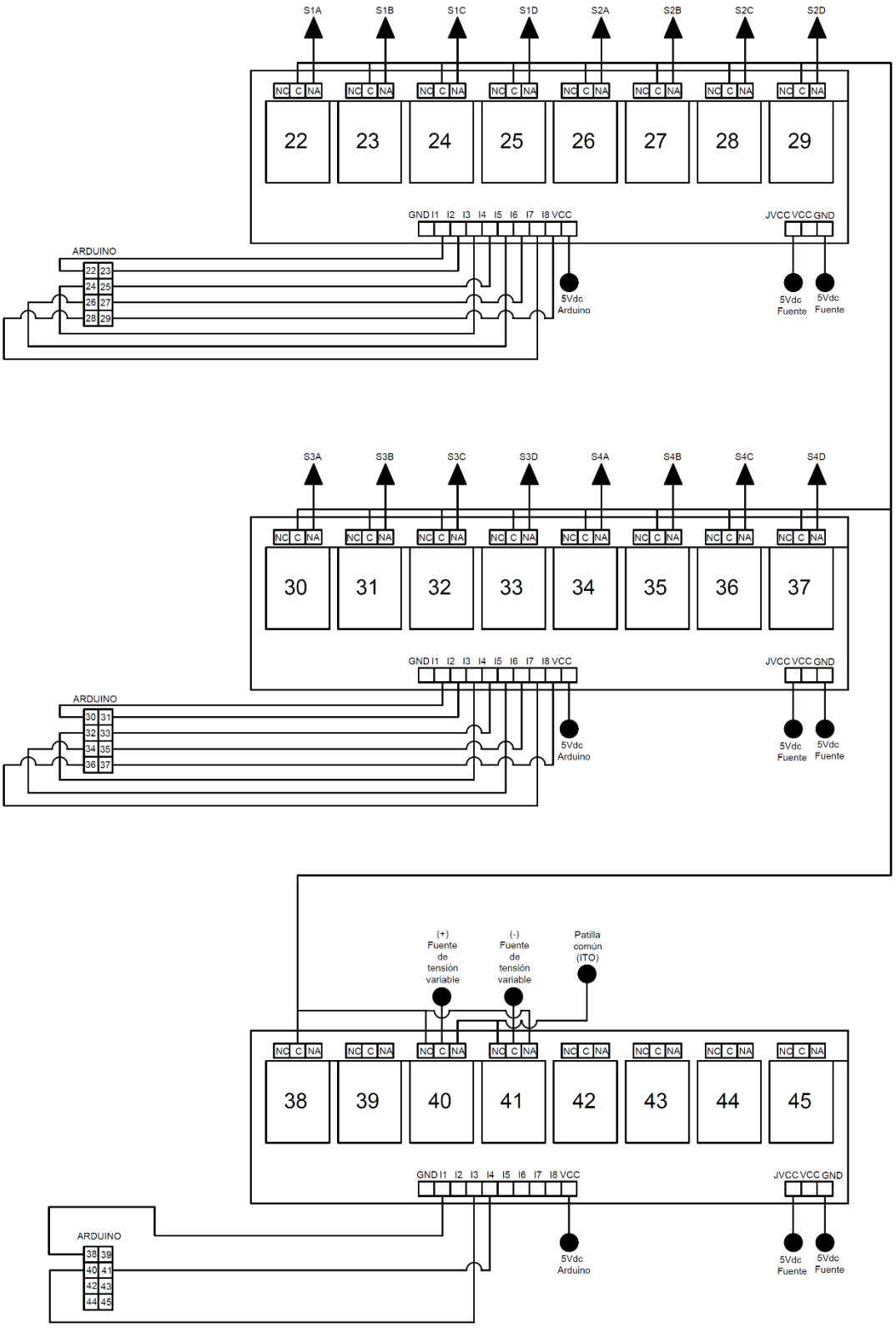
(11) http://www.seeedstudio.com/wiki/Grove_-_Temperature_and_Humidity_Sensor

(12) DHT11 Humidity & Temperature Sensor/ D-Robotics UK/ www.droboticsonline.com

(13) DHT11 Humidity & Temperature Sensor/ D-Robotics UK/ www.droboticsonline.com

9. ANEXOS

ANEXO 1: CONEXIONADO DE RELÉS.



	UNIVERSIDAD POLITÉCNICA DE CARTAGENA Proyecto Fin de Grado	Curso: 4º	Plano nº: 1
	Designación: Conexionado de relés	Tit.: GIE	