Robotización y control remoto de la cámara Mintron

P. de Vicente, J.A. López Pérez,
D. Cordobés, R. Bolaño, C. Almendros

Informe Técnico IT-OAN 2006-2

Índice

1. Introducción

Se ha adquirido una cámara CCD de la marca Mintron para el telescopio óptico que se pretende instalar en el subreflector del radiotelescopio del 40m. La cámara genera una señal de video compuesto y una señal de S-Video por sendos conectores. La cámara original dispone de cinco pulsadores situados en su parte posterior que permiten seleccionar diferentes modos de operación a través de un menú sobreimpresionado en la pantalla que se activa al pulsar prolongadamente el botón central. Dado que la cámara estará situada en el subreflector, un lugar de acceso complicado, se ha añadido un circuito que permite simular la pulsación de los botones y que permite su operación remota. El circuito, de pequeñas dimensiones, se ha instalado en el interior de la cámara.

Este informe describe las características de la cámara, el circuito y las modificaciones que se hicieron en ella, y el programa cargado en el microprocesador del circuito. Así mismo se describe el componente ACS que se ha escrito para controlar la cámara y un cliente en Python que se comunica con el componente.

2. Descripción del equipo

La cámara CCD es de la marca Mintron, modelo 12V1C-EX. La letra «C» en el modelo indica la versión PAL para Europa que utiliza 50 Hz y genera 625 líneas por cuadro. La cámara produce imágenes en blanco y negro. En su parte posterior dispone de cinco pulsadores (4 flechas y un botón central) que permiten controlar las opciones de configuración mediante un menú que aparece sobreimpresionado sobre la pantalla tras pulsar prolongadamente el botón central. Tiene dos salidas de video, una compuesta en un conector BNC y otra S-Video a través de un conector de 4 patillas. La alimentación de 12 V también está en un conector en su cara trasera. En la cara lateral derecha hay un conector para controlar remotamente una lente con auto iris. La CCD se encuentra en la cara frontal, donde hay un anillo con rosca hembra de 1/2 pulgada para poder roscar la cámara al telescopio o al sistema de enfoque. La figura 1 muestra una fotografía de la cámara en la que se ven las caras trasera, lateral derecha e inferior. La figura 2 muestra las dimensiones de la cámara. La tabla 1 resume las principales características de este modelo.

Según las especificaciones del fabricante las cámaras Mintron de la serie V (a la que pertenece esta) pueden detectar 0.0001 Lux si se les instala una CCD blanco y negro de 1/2 pulgada, como es el caso de este modelo. El modelo 12V1C-EX utiliza el chip ICX249AL de Sony de 1/2 pulgada. Este chip utiliza la tecnología «EX-View» que aumenta la sensibilidad de la CCD en un factor 2 para la luz visible y en un factor 4 en el infrarrojo cercano. Cada fotodiodo, correspondiente a un pixel, dispone de una lente microscópica fabricada para capturar y enfocar mejor la luz sobre la unión del semiconductor. La figura 3 muestra la respuesta espectral del chip.

3. Menú de control y características

El control de las opciones de la cámara se hace a través de un menú que se superpone sobre la imagen de vídeo y por el que se navega utilizando los pulsadores traseros de la cámara.

Figura 1: *Vista trasera de la cámara Mintron.*

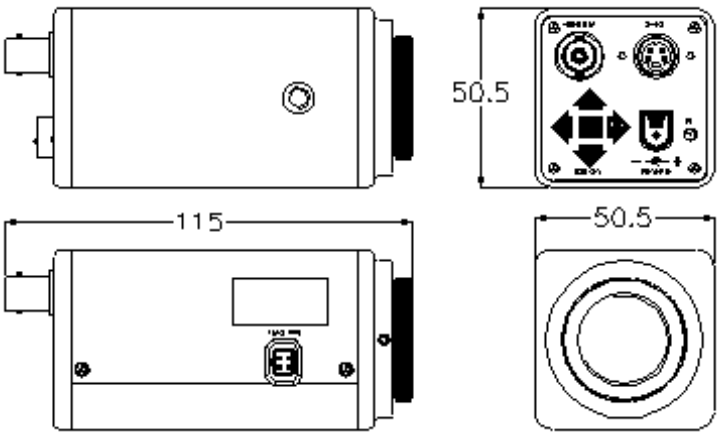

Figura 2: *Esquema y dimensiones de la cámara Mintron. Imagen superior izquierda: vista lateral derecha. Se aprecia el conector para el autoiris. Imagen superior derecha: vista trasera. Imagen inferior izquierda: vista lateral izquierda. Imagen inferior derecha: cara frontal.*

Propiedad	Valor
Modelo	12V1
Sistema de TV	CCIR
Sensor de imagen	Sensor de imagen CCD de 1/2-pulgada
Número total de pixels	795(H) $\times$ 596(V)
Sistema de barrido	625 líneas, 50 cuadros/segundo
Sistema de sincronismo	Interno / VD-Lock (opcional)
Iluminación mínima	Modo normal: 0,05 Lux (F1.2 , 5600K 30 IRE)
	Modo de luz estelar: 0,0005 Lux (F 0.8 , 5600K 10 IRE)
Resolución	600 LTV
Balance de blancos	Modo: ATW / AWC / FIX (Zero color rolling)
	Rango: 3200 - 10000 K (2200 - 15000 K con filtro S)
Control de la ganancia	Modo: AGC (ON / OFF)
	Rango: 0 - 18 dB
Relación señal/ruido	52dB (minima) / 60 dB (típica) (AGC OFF)
Diafragma electrónico	1/50 - 1/120,000 seg
Auto iris	A.E.S. / DC
Salida de video	Compuesto y salida Y/C 1.0V p-p a 75 ohmios
Corrección gamma	0,45
Temperatura de funcionamiento	-20 a 50 C
Humedad de funcionamiento	$>$ 85 %
Alimentación	Contínua 2V $\pm$1V / 180mA
Número de serie	H11058065
W/O	AF9400305

Cuadro 1: *Características más importantes de la cámara Mintron modelo 12V1-CX.*

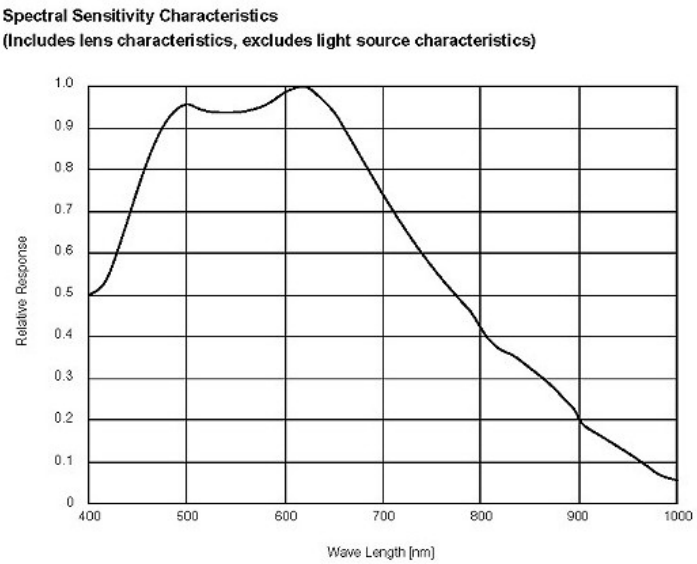

Figura 3: *Respuesta espectral del chip ICX249AL de Sony que equipa la cámara Mintron 12V1C-EX.*

El menú está compuesto por los siguientes elementos:

- `TITLE` Se puede insertar texto que se superpone sobre la imagen. Si `TITLE` está a `OFF` no se muestra ningún título en la pantalla. Para poder acceder al submenú del título es necesario que la opción derecha esté a ON. En caso contrario la pulsación del botón central no tiene efecto. Por omisión el título contiene puntos (·). El menú del título contiene letras mayúsculas, dígitos y algunos caracteres extra. Además permite moverse por el título a izquiera y derecha para cambiar selectivamente los caracteres situandose sobre la opción `SP->` o `<-SP`.

  La posición del título en la imagen está disponible a través del submenú `LOCATION`. Una vez dentro de este submenú las teclas de dirección permiten moverse por las esquinas para poder situar el título en cualquiera de ellas. Para salir de este submenú se pulsa el botón central.

- `SENSE UP` Permite seleccionar el tiempo de integración para el modo de función de luz estelar. Son posibles los siguientes valores: 1, 2, 4, 8, 12, 16, 24, 32, 48, 64 y 128 segundos. La sensibilidad crece a medida que se emplean tiempos de integración mayores. La imagen no se refresca hasta que la integración no se ha completado.

- `ALC/ELC` En este modo la cámara funciona en modo «normal». Es posible seleccionar la opción `ELC` (Electronic Light Control) o `ALC` (Autoiris Light Control) empleando las flechas de dirección derecha e izquierda. Para entrar en cada modo y configurar sus opciones es necesario seleccionarlo antes y pulsar el botón central después.

  En el modo `ELC` la cámara ajusta el nivel de luz automáticamente. No existe un obturador pero esta función emula su funcionamiento. Es posible fijar el brillo manualmente en

pasos de 1 en el intervalo [1,9]; valores más altos generan imágenes más claras.

En el modo ALC el ajuste del nivel de brillo se hace comandando remotamente un iris. Para que este modo sea operativo es necesario que haya un iris conectado cuya apertura sea controlable con corriente continua. Este menú dispone de un control de la velocidad del obturador y del nivel de brillo. Los posibles valores para la velocidad del obturador son: 50, 100, 120, 180, 250, 350, 500, 750, 1000, 1500, 2000, 3000, 4000, 6000, 8000 y 12000 s^{-1}. El nivel de brillo que se usa en este modo es el mismo que en el caso del modo ELC, de modo que este valor se comanda en cualquiera de los dos modos.

- AGC La CCD genera un voltaje que es función de la intensidad de corriente producida por los condensadores de la matriz. Este voltaje se transforma en una señal digital en un conversor AD. El conversor de esta cámara dispone de un control de la ganancia que ajusta el nivel de la señal de entrada. El control de la ganancia del menú permite fijar el comportamiento de la cámara Mintron de tres modos diferentes: automático, manual o desactivado. Para cambiar de modo se deben pulsar los botones de las flechas derecha o izquierda.

  Si AGC está a ON el control de la ganancia es automático de modo que el nivel máximo de la señal de vídeo sea 100 IRE o 720 mV. El control de la ganancia se hace sobre el conversor A/D. Se puede seleccionar un valor determinado de la ganancia, entrando en el submenú correspondiente y pulsando el botón central. El nivel es seleccionable entre 0 y 18 dB en pasos de 2,25 dB. El valor por omisión es el máximo. Con este modo se garantiza que el rango dinámico sea máximo en todo momento aunque en casos extremos el ruido se acrecienta.

  Si el modo es AGC Manual se fija el nivel de amplificación en un valor determinado antes de la digitalización. Al igual que en el caso anterior el nivel es seleccionable entre 0 y 18 dB en pasos de 2,25 dB. El valor por omisión es 0 dB. Si la cantidad de luz que llega a la CCD es muy pequeña o muy grande es posible que haya poco contraste.

  El modo AGC OFF es equivalente al modo manual con un nivel de ganancia de 0 dB.

- BLC Compensación de la luz de fondo. Esta opción no se ha incluido para ser configurada remotamente porque no estamos interesados en seccionar la imagen en campos pequeños y ajustar el nivel de luz en cada uno de ellos.

- W/B Balance de blancos. Esta opción no está operativa en este modelo en blanco y negro aunque aparentemente funciona en el menú.

- SYNC Opción no operativa porque es interna y no es seleccionable.

- OPTION Es un submenú que permite seleccionar diferentes opciones y al que se accede pulsando el botón central:

  - MASK Estas opciones están relacionadas con el uso de máscaras en la imagen. No tienen interés en astronomía por lo que no se comentan aquí.

- POSI/NEGA Este submenú permite «positivizar» o «negativizar» la imagen. Las imagenes negativas muestran oscuras las zonas más brillantes. Para alternar entre una opción y otra se deben usar los pulsadores de la izquierda o de la derecha.
- MIRROR Este submenú permite crear una imagen especular de la imagen original respecto de un eje vertical que divide la imagen en dos. Se alterna entre una posibilidad o la otra empleando el pulsador de la izquierda o el de la derecha.
- PRIORITY Este submenú permite controlar el modo de funcionamiento de la cámara en condiciones de poca visibilidad. Se pueden elegir dos modos SENSEUP o AGC.

  Si la prioridad es SENSEUP cuando el nivel de luz decrece la cámara comienza a aumentar el tiempo de integración desde 1 segundo hasta 128 para mantener el nivel de vídeo de 100 IRE. En caso de que el brillo sea tan pequeño que el nivel sea inferior a 100 IRE incluso con un tiempo de integración de 128 s entrará en funcionamiento el modo AGC haciendo la imagen más brillante y más ruidosa. Este modo es preferible en astronomía.

  Si la prioridad es AGC cuando el nivel de luz decrece la cámara pone en marcha el modo AGC para mantener el nivel de vídeo de 100 IRE. En caso de que se emplee el máximo de ganancia y el nivel sea insuficiente entra en funcionamiento el tiempo de integración desde 1 hasta 128 segundos. Este modo no es adecuado en astronomía porque aumenta el ruido de la imagen inmediatamente.

- ZOOM Este menú permite aumentar la imagen duplicando su tamaño. Como el zoom es digital en realidad sólo se consigue que la pantalla muestre una zona del campo de visión más pequeña en un tamaño mayor. Se puede seleccionar un zoom desde 1 hasta 2 en 8 pasos.
- SAVE Esta opción permite desactivar el menú de la pantalla y guardar las opciones de configuración incluso tras haber apagado la cámara. Usando los pulsadores izquierdo o derecho se puede elegir el modo alternativo PRESET que restituye los valores de fábrica. Esta operación tarda 3 segundos en completarse.

  En la tabla 2 se indican las opciones de configuración predefinidas (valores de fábrica):

4. Robotización de la cámara

El principal objetivo de este trabajo era modificar la cámara para poder operar sobre ella remotamente empleando un interfaz RS232 que emulara las pulsaciones sobre los botones en la parte posterior de la cámara. Se ha utilizado el documento denominado «Controlling the Mintron Camera remotely» como inspiración y como manual de la electrónica interna de la cámara, aunque no como guía ya que se ha implementado una solución diferente.

Propiedad	Valor por omisión
TITLE MODE	OFF
TITLE	. . . .
TITLE LOCATION	Esquina inferior izquierda
SENSE UP (función luz estelar)	OFF
ELC/ALC	ALC
ELC LEVEL	5
ALC LEVEL	5
ALC SHUTTER SPEED	OFF
BLC	OFF
AGC	OFF
AGC ON LEVEL	18 dB
AGC MANUAL LEVEL	0 dB
W/B	ATW
W/B MANUAL	3200 K
SYNC	INT
MASK A	OFF
MASK B	OFF
MASK C	OFF
MASK D	OFF
POSI/NEGA	POSI
MIRROR	OFF
PRIORITY	AGC
ZOOM	OFF

Cuadro 2: *Opciones de configuración predefinidas de la cámara Mintron modelo 12V1-CX.*

4.1. Selección del microcontrolador

En el Centro Astronómico de Yebes se tiene experiencia en la programación de microcontroladores de la firma MicroChip, en concreto del modelo PIC16F84A. Además, se dispone de herramientas para la compilación y descarga de código máquina en la memoria flash de los mismos. Por todo ello, se optó por el micro PIC16F84A, en lugar del propuesto en el documento guía de la firma Atmel, para los que no se dispone de dichas herramientas.

La hoja de características de este microcontrolador está disponible en www.microchip.com.

4.2. Circuito de control

La figura 4 muestra el circuito de control para la cámara Mintron. Si se compara con el utilizado en el documento guía, se observan dos diferencias fundamentales, amén del ya mencionado diferente controlador.

En primer lugar, se consigue la adaptación de los niveles de tensión RS-232 a TTL, compatible con la lógica digital del microcontrolador, mediante el circuito transmisor/receptor MAX3221. La solución propuesta en el documento guía es muy sencilla, pero no permite que el microcontrolador pueda devolver mensajes de respuesta tipo ACK.

En segundo lugar, la tensión de alimentación de 5 voltios del PIC se genera con ayuda del pequeño regulador 78L05 a partir de la tensión de entrada de 12 voltios de alimentación de la cámara, independizándola así del circuito de la cámara. Lo contrario ocurre en el circuito del documento guía, donde la alimentación de 3,3 voltios se toma de algún punto del circuito de la cámara.

La figura 5 muestra una fotografía de la tarjeta de control empotrada dentro de la cámara Mintron.

En la tabla 5 se detalla la lista de componentes utilizados:

Elemento	Cantidad	Referencia	Valor
1	5	C1,C2,C3,C4,C5	0.1 μF
2	1	C6	100nF
3	2	C7,C8	33pF
4	2	EMI2,EMI1	EMIFILTER
5	1	JP1	CONN PCB 6
6	1	JP2	CONN PCB 8
7	1	JS1	S-VHS
8	5	R6,R8,R10,R12,R14	6K8
9	5	R7,R9,R11,R13,R15	10K
10	1	U1	PIC16F84A
11	1	U2	MAX3221CAE/16SSOP
12	1	U3	LM78L05
13	1	X1	XTAL 4MHz

Cuadro 3: *Lista de componentes de la tarjeta de control.*

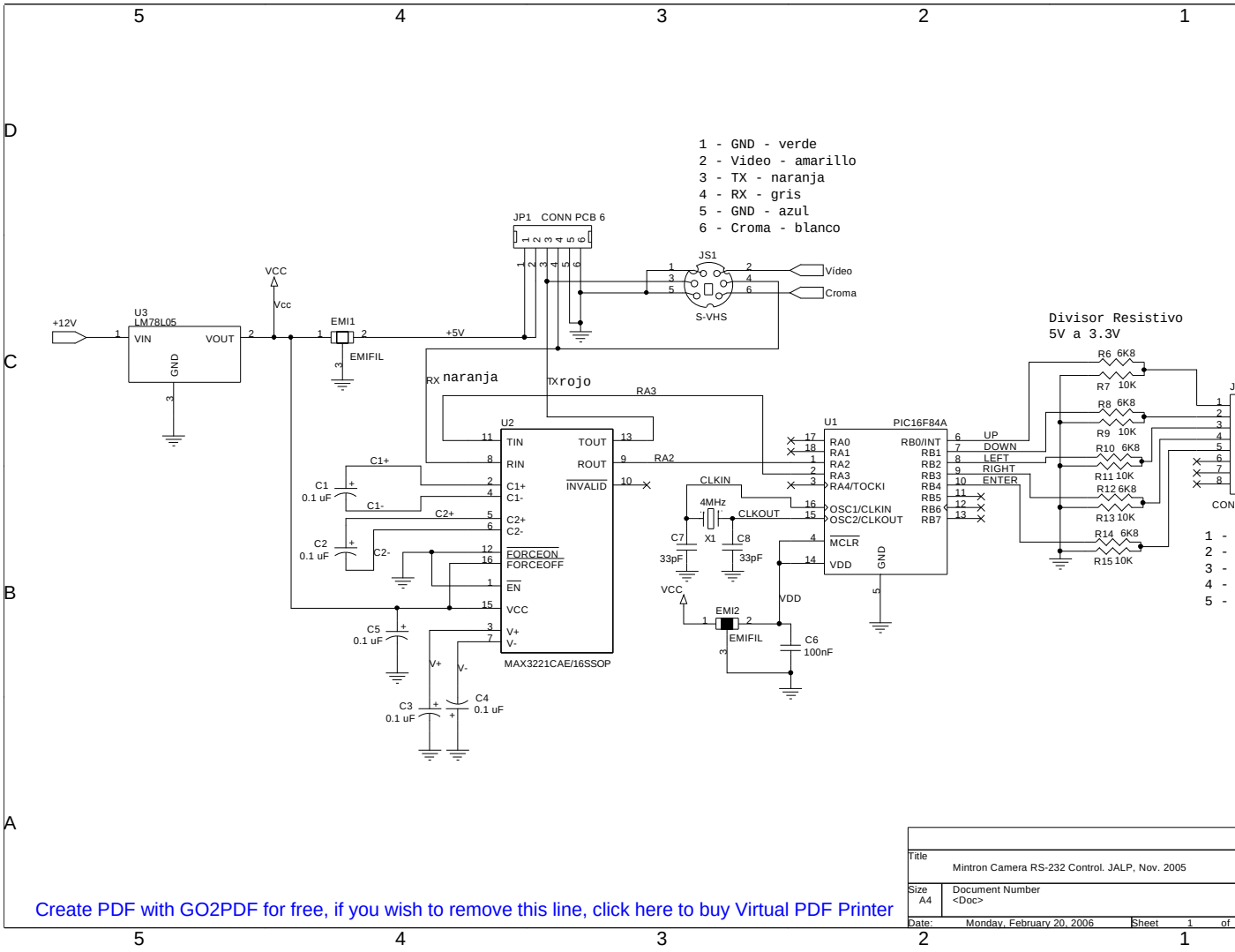

Figura 4: *Esquema eléctrico de la tarjeta de control remoto de la cámara Mintron.*

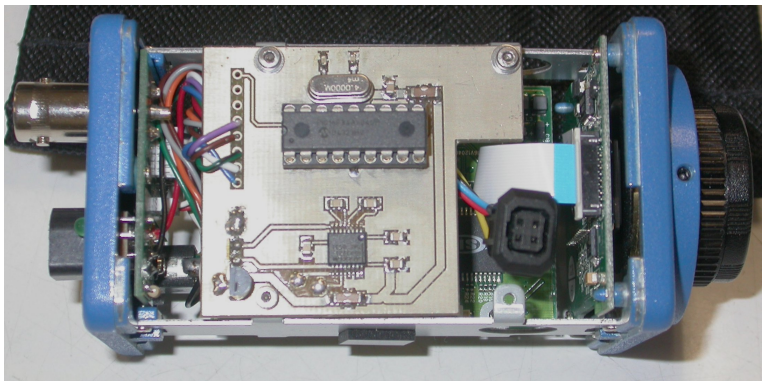

Figura 5: *Aspecto de la tarjeta de control remoto en el interior de la cámara Mintron.*

La conexión RS232 se hace a través de un conector miniDin hembra de cuatro patillas. Se ha reciclado un cable de un ratón PS2 en uno de cuyos extremos se conectó un terminal DB9. Las conexiones se detallan en la tabla **??**

Señal	Extremo DB9	Extremo miniDIN
TX	5	C1
RX	1	C6
GND	2	C7

Cuadro 4: *Lista de componentes de la tarjeta de control.*

4.3. Programación del microprocesador

El programa de control que ejecuta el microcontrolador fue escrito en lenguaje C con ayuda del compilador PCWH de la firma CCS. Una vez compilado, se genera el fichero HEX para su descarga en la memoria del microcontrolador.

El programa grabado en el microcontrolador consiste en un bucle infinito en el que en primer lugar se ponen las salidas a nivel alto, correspondiente al estado en el que los botones no están pulsados. Después se lee la entrada del microcontrolador conectada a la señal de recepción del conector RS232 y dependiendo del caracter leído en dicha entrada se ejecuta la función adecuada. Existen 6 funciones correspondientes a cada una de las pulsaciones de los botones (hay dos funciones para el botón central: pulsación corta y larga). Dentro de cada función lo primero que se hace es devolver una respuesta a través del puerto RS232 que coincide con el comando ejecutado y posteriormente se pone la salida de la patilla del PIC correspondiente a la pulsación de la tecla simulada a nivel bajo. Finalmente se hace un retardo de 100 ms en las pulsaciones cortas o de 1,5 segundos en las largas, con lo que se consigue mantener la tensión a nivel bajo durante dicho periodo de tiempo. Una vez finalizada la función comienza una nueva iteración del bucle principal. La tabla **??** muestra la lista de comandos disponibles, su respuesta, y el tiempo de demora hasta volver al bucle principal.

Orden	Respuesta	Demora (ms)
U ó 8	U	100
D ó 2	D	100
L ó 4	L	100
R ó 6	R	100
E ó 5	E	100
Q	Q	1500

Cuadro 5: *Ordenes que acepta el programa del microprocesador.*

El programa en lenguaje C ocupa 3 KB y una vez compilado 2 KB. El microcontrolador PIC16F84A impone una limitación en el tamaño del programa debido a su reducida memoria interna.

4.4. Proceso de grabación del programa

La grabación del programa en la memoria del microcontrolador, una vez compilado el código fuente, se llevó a cabo con un programador modelo MACH X de la firma CCS. Dicho programador se conecta al puerto USB de un PC.

La figura 6 muestra una fotografía del programador MACH X con el microcontrolador insertado en la posición correcta para su grabación.

Figura 6: *Programador MACH X.*

La primera vez que se conecta el programador el sistema operativo detecta el dispositivo y a continuación es necesario instalar el driver. Si omitimos este paso no se creará un vínculo de asociación entre el programador y el puerto USB y cuando ejecutemos el programa MACH X no será posible seleccionar la entrada adecuada a la que el programador está conectado. Hecho esto, ejecutamos el programa, seleccionamos el puerto al que está conectado el programador y ya es posible grabar el fichero HEX generado al compilar el código fuente.

5. Proceso de desarrollo del componente ACS

El proceso que se ha seguido en el desarrollo del componente es el mismo que con el resto de los componentes ACS para el sistema de control del radiotelescopio de 40m.

Se comenzó creando un directorio denominado `Mintron` con el subdirectorio *idl* y dos archivos: *mintronCamera.idl* y *mintronError.xml*. En el primero archivo se describe el interfaz de la cámara con sus propiedades y métodos y en el segundo archivo se definen las posibles excepciones que puede lanzar el componente en caso de error. Después se utilizó el generador de código `acsGenerator` para generar los ficheros de la implementación, las cabeceras y los DevIOs. Para poder generar las clases que heredan de DevIO el generador de código precisa

una un archivo auxiliar en el que figuren los nombre de las propiedades y su tipo. En nuestro caso el archivo `devionames` contenía las siguientes líneas:

```
title, DevIOtitle, ACE_CString
titlePosition, DevIOtitlePosition, mintronCamera::tMode
integrationTime, DevIOintegrationTime, CORBA::Double
elcLevel, DevIOelcLevel, CORBA::Long
shutterSpeed, DevIOshutterSpeed, CORBA::Long
gainAutomaticLevel, DevIOgainAutomaticLevel, CORBA::Double
gainManualLevel, DevIOgainManualLevel, CORBA::Double
zoomLevel, DevIOzoomLevel, CORBA::Double
```

Para lanzar el generador de código se introdujeron las siguientes ordenes en el intérprete del sistema operativo:

```
cd aries21/Mintron
acsGenerator --deviofile devionames idl/mintronCamera.idl
```

El generador de código crea los directorios `include`, `src` y `config`. A continuación en el subdirectorio `src` se desarrolló una clase C++ (`Mintron`) que implementa las funcionalidades de la cámara Mintron. La clase no es C++ pura porque los métodos pueden lanzar excepciones definidas en el archivo `mintronError.xml` y emplean la macro `ACS_LOG` para depurar el código.

El generador de código implementa el componente en numerosos archivos, uno por método, pero nosotros los reunificamos en uno sólo para evitar la dispersión de código. Los DevIOs se implementaron en el fichero de cabecera donde además se declaran, eliminando de este modo los archivos `.cpp` creados por el generador de código.

Se reformó el archivo `Makefile` para tener en cuenta todas las modificaciones posteriores, es decir, la ausencia de los archivos `.cpp` correspondientes a los DevIOs, la biblioteca para la clase C++ pura, y el uso de excepciones. Más abajo se muestra un extracto de la parte más importante del archivo `Makefile`:

```
# ....
#
# Includes (.h) files (public only)
# ---------------------------------
INCLUDES = mintronCameraImpl.h Mintron.h

#
# Libraries (public and local)
# ----------------------------
LIBRARIES = Mintron mintronCamera

Mintron_OBJECTS = Mintron
Mintron_LIBS = mintronError socketError sockscpp

#
# <brief description of mintronCamera library>
```

```
mintronCamera_OBJECTS =  mintronCameraImpl
mintronCamera_LDFLAGS =
mintronCamera_LIBS  = Mintron mintronCameraStubs mintronError
# .....
#
# Configuration Database Files
# ----------------------------
CDB_SCHEMAS = camera

#
# IDL Files and flags
#
IDL_FILES = mintronCamera
IDL_TAO_FLAGS =
USER_IDL =

#
ACSERRDEF = mintronError
```

Durante el proceso de compilación se genera la biblioteca del componente que carga el contenedor y la documentación en HTML empleando «doxygen». Si se mantienen actualizados los comentarios en el código la documentación, que se regenera automáticamente, permanece actualizada.

6. Descripción del componente ACS para la cámara

Se ha desarrollado una clase C++ que implementa la mayor parte de la funcionalidad de esta cámara. Dado que la cámara se controla a través de una conexión RS232, conectada a un conversor Lantronix serie-ethernet, en el constructor de la clase se pasa como parámetro la dirección IP del conversor Lantronix y el puerto en que se encuentra la cámara. No existe ningún método que permita variar estos valores sobre la marcha. Es necesario destruir el objeto y volver a crearlo con nuevos parámetros para realizar esta modificación. El constructor llama a un método llamado `resetLan()` que limpia el puerto al que se encuentra conectado la cámara para evitar interferencias en los comandos y en las lecturas a través de dicho puerto. La limpieza se consigue conectandose al puerto 23 del Lantronix y enviando la orden `logout port 2`. Para poder ejecutar esta orden es necesario entrar como usuario con todos los privilegios.

Existen 6 funciones básicas que emulan la pulsación de los botones: `upButton()`, `downButton()`, `rightButton()`, `leftButton()`, `enterButton()` y `enterButton2s()`. Estas funciones son privadas y sólo son accesibles a través de los métodos públicos.

El estado de la cámara se guarda en variables privadas. El contenido de estas variables se puede fijar a través de funciones públicas y se puede conocer a través de funciones «inline» públicas. La denominación que hemos seguido es del tipo `variable()` para recuperar el valor de una variable y `setVariable()` para fijar la variable y comandar el correspondiente modo en la cámara.

Las funciones que emulan las pulsaciones sobre los botones han requerido cierto ajuste ya que se debe guardar cierto tiempo entre dos pulsaciones consecutivas para conservar el sincronismo entre el microprocesador y el usuario. Se han investigado diferentes intervalos temporales y diversas formas de programar el microprocesador PIC. La solución elegida consiste en que el PIC recibe el comando e inmediatamente devuelve una respuesta, previa a cualquier operación interna. De este modo el retardo que se debe introducir es responsabilidad del programa que utiliza el usuario. El PIC, tras recibir la orden, devuelve la respuesta y aplica una tensión durante 100 ms para las pulsaciones normales y 1,5 segundos para la pulsación larga del botón central. Tras este intervalo quita la tensión y queda a la espera de un nuevo comando. Hemos encontrado que el mínimo tiempo admisible para una operación fluida es de 500 ms entre pulsaciones normales y 1,7 segundos tras una pulsación larga. Estos retardos se introducen en las funciones privadas que emulan las pulsaciones empleando la función *usleep()*. No hemos descubierto la razón por la que esto ocurre, pero parece estar relacionada con la velocidad del PIC y la grabación del comando y su extracción de su buffer.

También se ha descubierto que el posicionamiento del título no funciona si el título no contiene ningún carácter. Para evitar este problema el componente lanza una excepción en este caso. Además el posicionamiento del título no funciona adecuadamente tras varias repeticiones. Este comportamiento se ha descartado que proceda de una mala programación y se ha achacado al microprocesador. Dado que la cámara estará en un lugar difícilmente accesible se han implementado las funciones que pulsan los botones para permitir llevar la cámara manualmente a un estado adecuado en caso de un funcionamiento inadecuado de esta. En una primera versión estos métodos no estaban accesibles y formaban parte de las funciones privadas de la clase en C++. Tan sólo se ha dejado como función privada la pulsación prolongada del botón central para impedir que los usuarios puedan acceder arbitrariamente al menú sin pasar por las propiedades.

A continuación se incluye una relación de las funciones de la clase `Mintron` y de sus variables privadas:

Funciones públicas:

`void Mintron(const char *`**tcpiaddress**`, unsigned int` **port**`)`

Constructor de la clase: reinicia el dispositivo Lantronix y abre un conexión TCP con la dirección y el puerto que se pasan como parámetro.

tcpiaddress Dirección del dispositivo Lantronix.

port *Puerto [1,4] al que está conectado la cámara.*

`void resetLan (const char *`**tcpiaddress**`, unsigned int` **port**`) throw (mintronError::cannotConnectExImpl &, mintronError::writeErrorExImpl &)`

Reinicia el dispositivo Lantronix abriendo una sesión al puerto 23 y reiniciando el puerto al que está conectada la cámara.

tcpiaddress Dirección del dispositivo Lantronix

port *Puerto [1,4] que se reinicia.*

void setTitle (const char *__name__) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &)

La cámara Mintron permite superponer un título de 12 caracteres sobre la imagen.

name Título a escribir en la pantalla

char * title (void)

La cámara Mintron permite superponer un título de 12 caracteres sobre la imagen.

name Título en la pantalla

void setTitlePosition (unsigned int **position**) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &, mintronError::paramOutOfLimitsExImpl &)

La cámara Mintron permite superponer un título de 12 caracteres sobre la imagen. Este procedimiento permite situar el título en cualquiera de las cuatro esquinas de la pantalla.

position Esquina inferior izquierda: 0, esquina inferior derecha: 1, esquina superior izquierda: 2, esquina superior derecha 3

unsigned int titlePosition (void)

La cámara Mintron permite superponer un título de 12 caracteres sobre la imagen. Este procedimiento devuelve la posición del título en la pantalla.

position La posición del título en la pantalla. Esquina inferior izquierda: 0, esquina inferior derecha: 1, esquina superior izquierda: 2, esquina superior derecha 3

void setTitleMode (bool **mode**) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &)

La cámara Mintron permite superponer un título de 12 caracteres sobre la imagen. Este procedimiento activa o desactiva el título.

mode Si verdadero, el título se activa, si falso se desactiva.

bool titleMode (void)

La cámara Mintron permite superponer un título de 12 caracteres sobre la imagen. Este procedimiento devuelve el estado del título.

Devuelve verdadero si el título está activo, falso si no lo está.

void reset (void) throw (mintronError::readErrorExImpl &, mintron-Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &)

Fija las opciones de configuración a los valores predefinidos.

void setIntegrationTime (unsigned int **intTime**) throw (mintronError::read-ErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrong-AcknowledgementExImpl &, mintronError::paramOutOfLimitsExImpl &)

Selecciona el tiempo de integración máximo para la función de luz estelar.

intTime Tiempo de integración en segundos.

unsigned int integrationTime (void)

Devuelve el tiempo de integración seleccionado para la función de luz estelar.

void setLightControlMode (bool **mode**) throw (mintronError::readError-ExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgeme ExImpl &)

Alterna el modo control de la luz: ELC (Electronic Light Control) o (ALC) Autoiris Light Control)

mode Modo de control de la luz

bool lightControlMode (void)

Devuelve el modo de control de la luz: verdadero si el modo es ELC y falso si es ALC

void setShutterSpeed (unsigned int **speed**) throw (mintronError::read-ErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrong-AcknowledgementExImpl &, mintronError::paramOutOfLimitsExImpl &)

Fija la velocidad de la lente del autoiris cuando la cámar Mintron está en modo ALC.

speed Velocidad del obturador

unsigned int shutterSpeed (void)

Devuelve la velocidad del obturador del autoiris cuando la cámar Mintron está en modo ALC.

void setElcLevel (unsigned int **level**) throw (mintronError::readError-ExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgeme ExImpl &, mintronError::paramOutOfLimitsExImpl &)

Fija el brillo utilizado en modo ELC

level *Nivel usado para el ELC, valor entero dentro del intervalo [1,9]*

unsigned int elcLevel (void)

Devuelve el brillo utilizado en el modo ELC.

void setAGCMode (unsigned int **mode**) throw (mintronError::readError-ExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgeme ExImpl &)

Fija el modo del control de la ganancia.

mode Si vale 0, AGC están en ON, si vale 1 está en MANUAL y si vale 2 está OFF

unsigned int AGCMode (void)

Devuelve el modo del control de la ganancia. Si vale 0, AGC están en ON, si vale 1 está en MANUAL y si vale 2 está OFF

void setManualAGCLevel (double **level**) throw (mintronError::readError-ExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgeme ExImpl &, mintronError::paramOutOfLimitsExImpl &)

Fija la ganancia para el modo automático (AGC ON)

Parámetro: **level** *Nivel en dB. Puede tomar valores entre 0 y 18 dB en pasos de 2,25 dB.*

void setAutomaticAGCLevel (double **level**) throw (mintronError::read-ErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrong-AcknowledgementExImpl &, mintronError::paramOutOfLimitsExImpl &)

Fija la ganancia para el modo manual (AGC MANUAL)

Parámetro: **level** *Nivel en dB. Puede tomar valores entre 0 y 18 dB en pasos de 2,25 dB.*

double manualAGClevel (void)

Devuelve el nivel en Db usado cuando el control de ganancia es manual

double automaticAGClevel (void)

Devuelve el nivel en Db usado cuando el control de ganancia es automático

void setMirrorMode (bool **mode**) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgement-ExImpl &)

Activa o desactiva el modo especular en la imagen a lo largo de un eje de simetría vertical.

Parámetro: **mode** *Si verdadero genera la imagen especular de la imagen, si falso mantiene la imagen original*

`bool mirrorMode (void)`

Devuelve el estado de inversión de imagen. Si verdadero la imagen está invertida, si falso no lo está

`void setNegativeMode (bool` **mode**`) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &)`

Negativiza o positiviza la imagen

mode Parámetro: **mode** *Si verdadero la imagen se negativiza, si falso se mantiene en positivo (valor original)*

`bool negativeMode (void)`

Devuelve el estado de la negativización de la imagen. Si verdadero la imagen está negativizada. Si falso la imagen es la original.

`void setPriority (bool` **senseupMode**`) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgemeExImpl &)`

Establece la prioridad en el uso bajo condiciones de poca luz primero AGC o tiempo de integración

Parámetro: **senseupMode***. Si verdadero la prioridad es el tiempo de integración, si falso se usa el control autromático de la ganancia*

`bool priority (void)`

Devuelve la prioridad en condiciones de poca visibilidad. Verdadero implica el uso del tiempo de integración, falso el control automático de la ganancia.

`void setZoomMode (bool` **zMode**`) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgement-ExImpl &)`

Activa o desactiva el zoom digital.

zMode Si verdadero activa el zoom digital, si falso lo desactiva

`bool zoomMode (void)`

Devuelve el estado del zoom. Si verdadero el zoom está activado, si falso está desactivado

```
void setZoom (double value) throw (mintronError::readErrorExImpl &,
mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgement-
ExImpl &, mintronError::paramOutOfLimitsExImpl &)
```

Fija el valor del zoom digital

Parámetro: **value** *Zoom digital. Puede tomar cualquier valor enel intervalo [1,2] en pasos de 0,125.*

```
double zoom (void)
```

Devuelve el zoom digital en uso

```
void enterButton (void) throw (mintronError::readErrorExImpl &, mintron-
Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl
&)
```

Emula la pulsación del botón central durante 100 ms y una espera de 400 ms más.

```
void upButton (void) throw (mintronError::readErrorExImpl &, mintron-
Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl
&)
```

Emula la pulsación del botón superior durante 100 ms y una espera de 400 ms más.

```
void downButton (void) throw (mintronError::readErrorExImpl &, mintron-
Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl
&)
```

Emula la pulsación del botón inferior durante 100 ms y una espera de 400 ms más.

```
void rightButton (void) throw (mintronError::readErrorExImpl &, mintron-
Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl
&)
```

Emula la pulsación del botón derecho durante 100 ms y una espera de 400 ms más.

```
void leftButton (void) throw (mintronError::readErrorExImpl &, mintron-
Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl
&)
```

Emula la pulsación del botón izquierdo durante 100 ms y una espera de 400 ms más.

```
void enterTitleMenu (void) throw (mintronError::readErrorExImpl &,
mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgement-
ExImpl &)
```

Procedimiento que permite entrar en el menú del título cuando el menú principal no está activado

Funciones privadas:

void enterButton2s (void) throw (mintronError::readErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrongAcknowledgement-ExImpl &)

Emula la pulsación del botón central durante 1,5 segundos y una espera de 200 ms más.

void activateMenu (void) throw (mintronError::readErrorExImpl &, mintron-Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &)

Enciende el menú principal y lo superpone en la pantalla

unsigned levelPosition (double **level**) throw (mintronError::paramOut-OfLimitsExImpl &)

Devuelve el número de pulsaciones necesarias para fijar el valor que se pasa. Si el nivel no está en la lista interna se genera una excepción. Esta función se emplea para fijar el nivel empleado por el ELC y el ALC.

level Nivel de luz a usar en unidades arbitrarias.

unsigned int timePosition (unsigned int **itime**) throw (mintronError::param-OutOfLimitsExImpl &)

Devuelve el número de pulsaciones necesarias para fijar el valor que se pasa. Si el valor no está en la lista interna se genera una excepción. Esta función se emplea para fijar el tiempo de integración.

level Tiempo de integración a usar en segundos.

void enterAGCsubmenu (unsigned int **mode**) throw (mintronError::read-ErrorExImpl &, mintronError::writeErrorExImpl &, mintronError::wrong-AcknowledgementExImpl &, mintronError::paramOutOfLimitsExImpl &)

Activa el menú principal y entra en el submenú AGC correspondiente.

mode. Si vale 0 el modo es AGC ON, si vale 1 el modo es AGC MANUAL, si vale 2 el modo es AGC OFF.

void exitAGCsubmenu () throw (mintronError::readErrorExImpl &, mintron-Error::writeErrorExImpl &, mintronError::wrongAcknowledgementExImpl &)

Sale del submenú AGC y desactiva el menú principal.

`unsigned int zoomPosition (double` **zoom**`) throw (mintronError::paramOutOfLimitsExImpl &)`

Devuelve el número de pulsaciones necesarias para fijar el valor que se pasa. Si el valor no está en la lista interna se genera una excepción. Esta función se emplea para fijar el zoom digital

zoom Valor del zoom digital

`unsigned int speedPosition (unsigned int` **speed**`) throw (mintronError::paramOutOfLimitsExImpl &)`

Devuelve el número de pulsaciones necesarias para fijar el valor que se pasa. Si el valor no está en la lista interna se genera una excepción. Esta función se emplea para fijar la velocidad del obturador.

zoom Velocidad del obturador

`void defaultValues (void)`

Fija los valores de las variables privadas a su valor predefinido, que coincide con los de fábrica y que se obtienen al reiniciar la cámara.

`void flushBuffer ()`

Lee el flujo de caracteres de un socket hasta que no queda ninguno en el «buffer»

La figura 7 muestra un esquema de las clases empleadas.

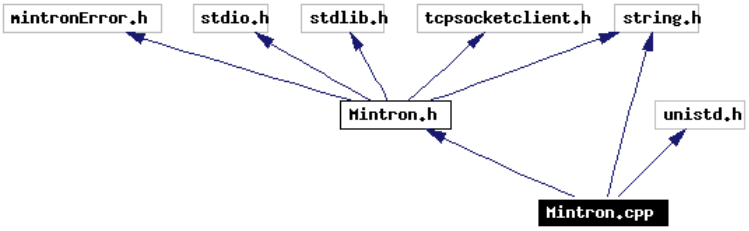

Figura 7: *Gráfico de dependencias de la clase Mintron*

- Todos los comandos que necesitan modificar un valor numérico dejan el modo de la cámara en su estado anterior.
- Los comandos se copian a un buffer antes de enviarlos empleando la instrucción `sprintf()`.
- Se han incluido instrucciones con ACS_SHORT_LOG en las funciones emuladoras de los pulsadores para depurar el envío de comandos a la cámara.

- El tiempo de integración elegible tiene que ser igual a cualquiera de los valores de la lista, de lo contrario la clase devuelve una excepción del tipo «paramOutOfLimits».
- Los posibles valores del tiempo de integración son: 1, 2, 4, 8, 12, 16, 24, 32, 48, 64 y 128 s.
- La velocidad del obturador para un auto iris tiene que ser igual a cualquiera de los valores de la lista, de lo contrario la clase devuelve una excepción del tipo «paramOutOfLimits».
- Los posibles valores de la velocidad del obturador son: 50, 100, 120, 180, 250, 350, 500, 750, 1000, 1500, 2000, 3000, 4000, 6000, 8000 y 12000 s^{-1}.
- El nivel del control electrónico de la luz tiene que ser igual a cualquiera de los valores de la lista, de lo contrario la clase devuelve una excepción del tipo «paramOutOfLimits».
- Los posibles valores del nivel del control electrónico de la luz son cualquier valor entero en el intervalo [1,9].
- El control de la ganancia (automática o manual) tiene que ser igual a cualquiera de los valores de la lista, de lo contrario la clase devuelve una excepción del tipo «paramOutOfLimits».
- Los posibles valores del control de la ganancia (automática o manual) son cualquier valor entero en el intervalo [1,9].
- El zoom elegido tiene que ser igual a cualquiera de los valores de la lista, de lo contrario la clase devuelve una excepción del tipo «paramOutOfLimits».
- Los posibles valores del zoom son: 1, 1,125, 1,250, 1,375, 1,50, 1,675, 1,750, 1,875 y 2

6.1. Errores. Lanzamiento de excepciones

En el componente ACS de la cámara Mintron se han definido un conjunto de excepciones en el fichero `mintronCameraError.xml` y se han identificado con el número 2007. En la lista que se relaciona a continuación se incluye el nombre identificativo y la descripción que figura en el archivo `mintronCameraError.xml`:

mintronCorrect Description: «Mintron request sucessfully procesed»

cannotConstruct Description: «Mintron constructor error»

cannotConnect Description: «Cannot connect to Mintron camera»

cannotInitialize Description: «Cannot talk and initialize the Mintron camera». Se produce cuando se produce un error en la conexión de los sockets.

writeError Description «Error sending data to Mintron camera». Esta excepción se genera a partir de una excepción en la escritura sobre un objeto de la clase *Tcpsockclient*

readError Description «Error receiving data from Mintron camera». Esta excepción se genera a partir de una excepción en la lectura de un objeto de la clase *Tcpsockclient*

paramOutOfLimits Description: «Parameters sent to Mintron out of limits»

wrongAcknowledgement Description: «Wrong acknowledgement from Mintron»

wrongMode Description: «Cannot read/write property from this mode. Activate corresponding mode first»

Las excepciones se lanzan desde la clase C++ pura, desde las clase de los DevIOs y desde la clase que implementa el componente, de acuerdo con la política de ACS.

6.2. Definición del archivo IDL

En el archivo IDL se ha definido un módulo llamado *mintronCamera* y un interfaz denominado *camera* que hereda de *ACS::CharacteristicComponent*. En el módulo se han definido varios ENUMs que se utilizan para describir el estado de la cámara Mintron y a partir de los cuales se han creado varias propiedades ACS. Sólo se ha definido un método: *reset()*, que se utiliza para reiniciar la cámara y fijar sus opciones a los valores predefinidos. La cámara se controla a través de propiedades de lectura y escritura que en algunos casos son enums, enteros largos (long) y reales largos (double). Existen dos propiedades de sólo lectura: la dirección IP del dispositivo Lantronix y el puerto al que se conecta el equipo que se leen durante la inicialización a partir del archivo XML en la base de datos.

Las propiedades del interfaz se relacionan a continuación:

ACS::ROstring lantronixIPNumber: Número TCIP en forma de cadena del dispositivo Lantronix al que está conectada la cámara.

ACS::ROlong lantronixSerialPort: Puerto serie del dispositivo Lantronix al que está conectada la cámara.

ACS::RWstring title: Título que se desea sobreimpresionar sobre la pantalla.

RWtPosition titlePosition: Posición del título en la imagen. Se trata de un «enum».

RWtMode titleMode: Modo del título: activado o desactivado.

ACS::RWlong integrationTime: Tiempo de integración en segundos.

RWlightCtlMode LightControlMode: Modo de control de la luz: ELC o ALC.

ACS::RWlong elcLevel: Nivel de brillo empleado en el modo ELC.

ACS::RWlong shutterSpeed: Velocidad del obturador del autoiris (s^{-1} en el modo ALC.

RWagcMode gainMode: Modo de ganancia: automático, manual o desactivado.

ACS::RWdouble gainAutomaticLevel: Nivel de ganancia en dB cuando el control es automático.

ACS::RWdouble gainManualLevel: Nivel de ganancia en dB cuando el control es manual.

RWnegMode negativeMode: Inversión de la imagen: positivada o negativizada.

RWmirrMode mirrorMode: Inversión de la imagen con un eje de simetría vertical.

RWzMode zoomMode: Modo de zoom digital: activado o desactivado.

ACS::RWdouble zoomLevel: Grado de zoom. Puede tomar valores entre 1 y 2.

RWsenseUpMode priorityMode: Modo de prioridad: AGC o tiempo de integración.

6.3. Implementación del componente: servidor

La implementación se hace a través de los métodos de lectura y escritura de las clases de los DevIO; sólo existe un método `reset()`. En el constructor de cada clase DevIO se pasa una referencia del objeto *Mintron*. De este modo todos los métodos públicos de esta clase están disponibles al DevIO. Los métodos de lectura y escritura crean un *muttex* para impedir la modificación de las variables por varios usuarios simultáneamente, llaman al método necesario de la clase Mintron y actualizan el valor de la variable privada. El *muttex* también se emplea en el método `reset()`.

La figura 8 muestra un esquema de la interrelación entre la clase de la implementación (*cameraImpl*), las clases de los DevIOs y la clase *Mintron*:

La clase *cameraImpl* instancia los DevIOs en el método initialize() en lugar de hacerlo en el constructor, ya que, en aquel, habitualmente se reserva memoria o/y se abre una conexión con un socket. Cualquiera de las operaciones anteriores puede fallar y resulta más adecuado en ese caso que estas operaciones se hagan en la inicialización del componente según la política de ACS. Al crearse cada DevIO, se lee su valor predeterminado de la base de datos (del archivo XML) y se llama al método *write()* que escribe el valor de dicha variable. Para impedir que la inicialización del componente tarde 1 minuto 50 segundos (tiempo empleado en recorrer el menú inicializando propiedades), el constructor de la clase C++ *Mintron* fija el valor predefinido de las variables internas de acuerdo con el valor que se obtiene al reiniciar la cámara. Los métodos sólo activan el menú de operaciones si existe una diferencia entre el valor de cada variable interna y el valor comandado. De este modo se reduce el tiempo de inicialización a 20 segundos.

Existe una diferencia entre los valores inicializados por el componente de la cámara y los valores predefinidos de esta. El método `reset()` deja la cámara en su modo predefinido, pero el componente al inicializarse difiere de este estado en dos propiedades. El modo es ELC (el predefinido es ALC) y el título (predifinido en blanco) que contiene la cadena OAN.

6.4. Implementación del cliente en Python

Se ha escrito un cliente gráfico en Python para el control de la cámara. En primer lugar se empleó el diseñador gráfico de Qt que guarda el aspecto gráfico en un archivo XML que

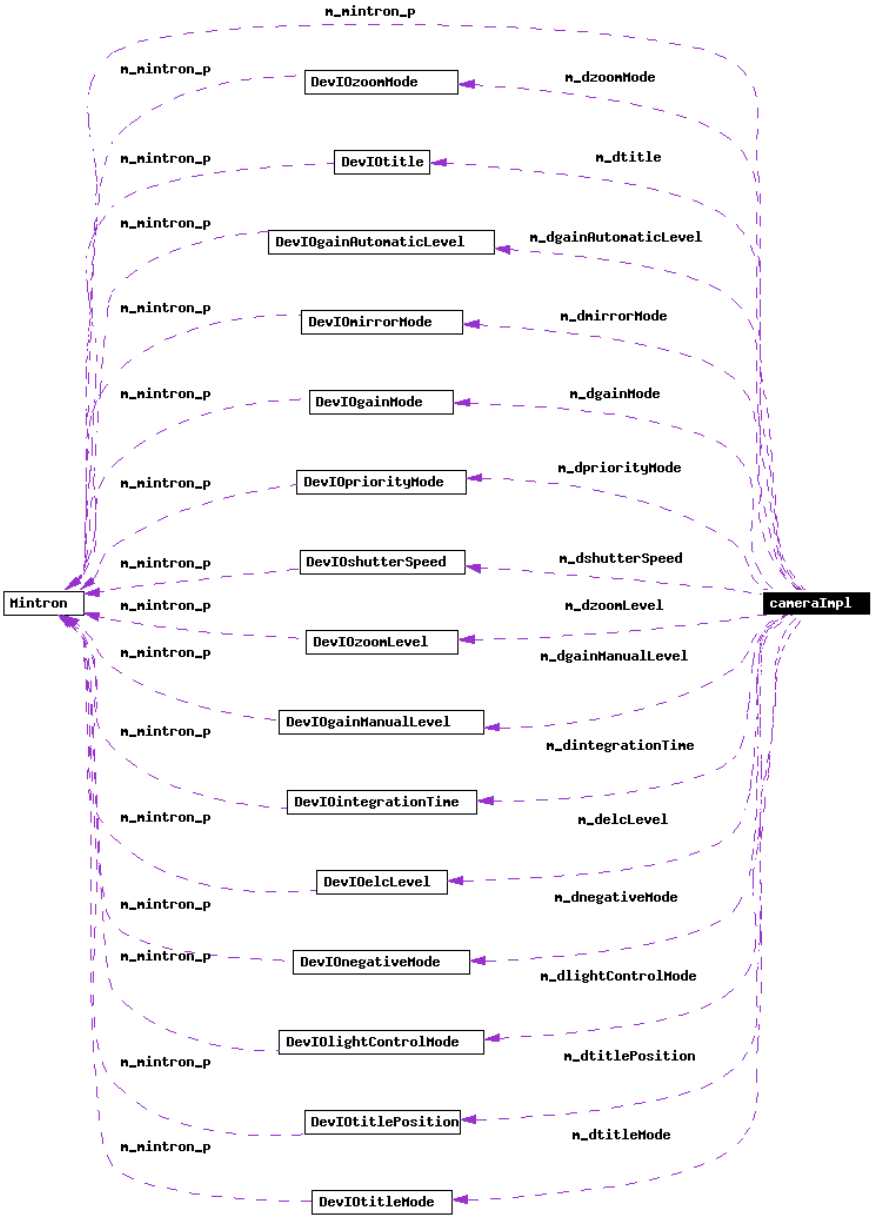

Figura 8: *Gráfico de relaciones entre la implementacion, los DevIOs y la clase Mintron*

se denominó `mintronUI.ui`. A continuación se empleó el programa `pyuic` para generar una clase en Python (`mintronUI.py` que emplea los paquetes de Qt para Python. Finalmente se escribieron dos clases más:

- `mintronACS` reside en el fichero `pythonClient.py`. El constructor de esta clase crea un objeto de la clase `PySimpleClient`, otro del componente «MINTRON» y uno por cada propiedad de este componente:

```
class mintronACS:
   def __init__(self,argv):
      self.__sc = PySimpleClient()
      self.__mintron = self.__sc.getComponent("MINTRON")

      self.__lantronixIPNumber = self.__mintron._get_lantronixIPNumber()
      self.__lantronixSerialPort = self.__mintron._get_lantronixSerialPort()
      .....
```

  Además en esta clase se definen métodos para fijar y leer las propiedades. Por ejemplo:

```
    ....
    def setIntegrationTime(self, value):
        """Sets the integration time.
        """
        self.__titleMode.set_sync(value)

    def integrationTime(self):
        """@return integration time.
        """
        return (self.__integrationTime.get_sync()[0])
      .....
```

- `mintronGUI` reside en el fichero `mintronGUI.py`. Esta clase hereda de `mintronUI` y por tanto tiene acceso a todos los «widgets» gráficos. En el constructor de la clase se crea un objeto de la clase `mintronACS` y de este modo se tiene acceso completo a todos sus métodos:

```
class mintronGUI(mintronUI):
    def __init__(self):
        mintronUI.__init__(self)
        self.__mintron = mintronACS("")
        ....
```

  Finalmente en esta clase se asocian los eventos gráficos a los métodos de la clase `mintronACS` y se inicializan los valores de los «widgets» tras leer el estado del componente a través de los métodos de lectura de las propiedades definidos en `mintronACS`.

La figura 9 muestra el cliente gráfico definitivo. Aquellas propiedades que sólo admiten ciertos valores no consecutivos utilizan una lista desplegable para que el usuario puede seleccionar la que precise. Sólo se ha utilizado un deslizador más un «spinbox» para controlar una propiedad cuyos valores son consecutivos permanecen dentro de un intervalo. La selección de los diferentes modos se hace a través de botones de selección excluyente, conocidos habitualmente como botones «radio». Finalmente se ha incluido un botón para restablecer los valores de fábrica de la cámara.

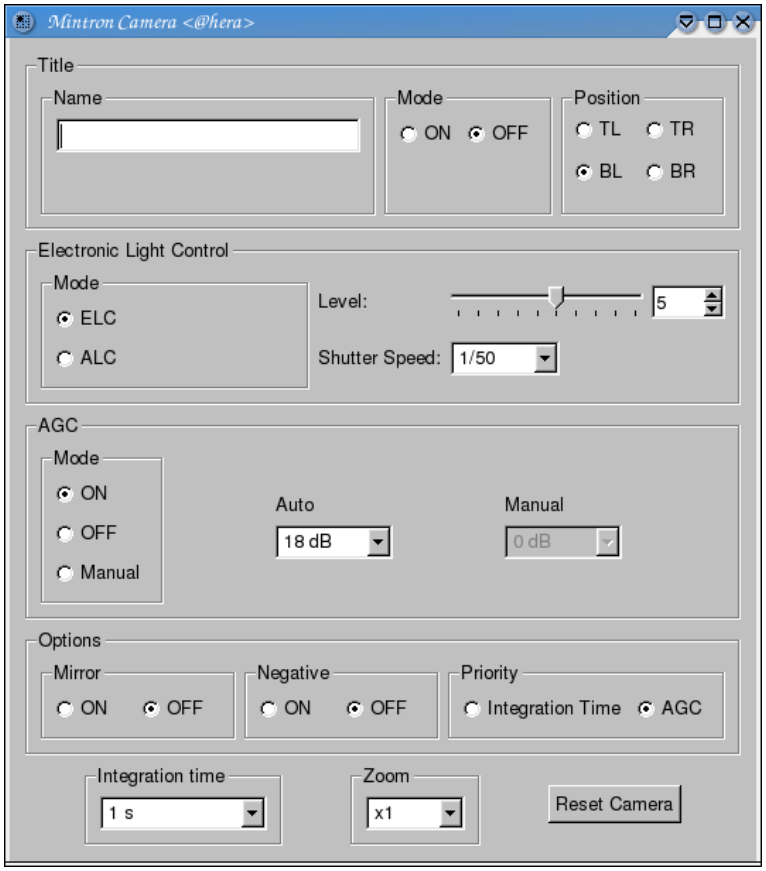

Figura 9: *Captura del cliente gráfico en Python*

Dado que cada instrucción puede tardar varios segundos en completarse, el cliente no se refresca hasta que esta se completa. Si durante ese tiempo se desplaza una ventana por encima de la del cliente y luego se retira, éste permanecerá en gris hasta que la instrucción se complete.

Referencias

[1] “http://mintron.com.tw/product/spectrum/SPECTRUM.htm”

[2] “http://www.mintron.com/HTM/New_PRODUCTS/ALPHA/SPEC/12v1.htm”

[3] “http://mintron.com.tw/product/spectrum/Sony/ICX249AL.html”

[4] . Bopp, 2004, “Controlling the Mintron Camera remotely”