DEPARTAMENTO DE ELECTRONICA Y TELECOMUNICACIONES
ESCUELA UNIVERSITARIA DE INGENIERIA
VITORIA - GASTEIZ
UNIVERSIDAD DEL PAIS VASCO - EUSKAL HERRIKO UNIBERTSITATEA

Ampliación de Sistemas Digitales

Lector de monedas

José Miguel Gil-García
Dpto. Electrónica y Telecomunicaciones
E.U.I. de Vitoria-Gasteiz
Enero 2006

Lector de monedas

Objetivos

Los objetivos de este cuaderno son los siguientes:

- Comprender el funcionamiento de dispositivos capaces de detectar y discriminar el valor de monedas
- Diseñar funciones no bloqueantes para la lectura de periféricos lentos.

Introducción

Se dispone de un lector de monedas de referencia C120 de la empresa Coin Controls cuyo funcionamiento se detalla a continuación.

Funcionamiento del detector de monedas C120

El validador o detector de monedas C120 es capaz de discernir entre seis tipos de monedas que previamente tenga almacenadas en memoria. Dispone de un conector para cable plano de 10 vías con la siguiente disposición

Pin	Descripción	Activo con	P0, P1 y P2 en 51
1	GND		
2	12V DC		
3	Accept 5 (O)	Bajo	PX.0
4	Accept 6 (O)	Bajo	PX.1
5	Return (O)		PX.2
6	Inhibir todas (O)	Alto	PX.3
7	Accept 1 (O)	Bajo	PX.4
8	Accept 2 (O)	Bajo	PX.5
9	Accept 3 (O)	Bajo	PX.6
10	Accept 4 (O)	Bajo	PX.7

Tabla 1 Distribución del conector de interfaz del detector de monedas

Cada una de las salidas del detector es de tipo drenador abierto y es capaz de absorber una corriente máxima de 100mA. Las entradas son compatibles TTL. La salida Return se activa cuando

el usuario presiona el botón de devolución de moneda. Cuando se introduzca una moneda tenida por válida por el detector se activará (pondrá a cero) durante un tiempo de 100ms ±20% el pin correspondiente según se haya programado de fábrica tal y como se muestra en la Fig. 1. La señal de Inhibir Todas (Inhibit all) activa a nivel alto evita que el detector lea más monedas y las rachaza todas, sean del valor que sean.

Fig. 1 Forma de onda de la señal de salida del detector de monedas

El detector trae un DIP switch situado bajo la tapa con el que se pueden inhibir de forma individual la aceptación de uno de los tipos predeterminados de moneda. La programación que se haga mediante este método quedará de forma permanente y no podrá ser controlada por el firmware del micro que se encargue de leer el detector. Cada moneda tendrá su equivalente pestaña en el DIP Switch para ser inhibida (pestaña hacia arriba) o aceptada (pestaña hacia abajo), siempre y cuando la entrada digital *Inhibit all* esté a nivel bajo.

El camino que sigue la moneda en función de si se acepta o se rechaza queda descrito en la Fig. 2

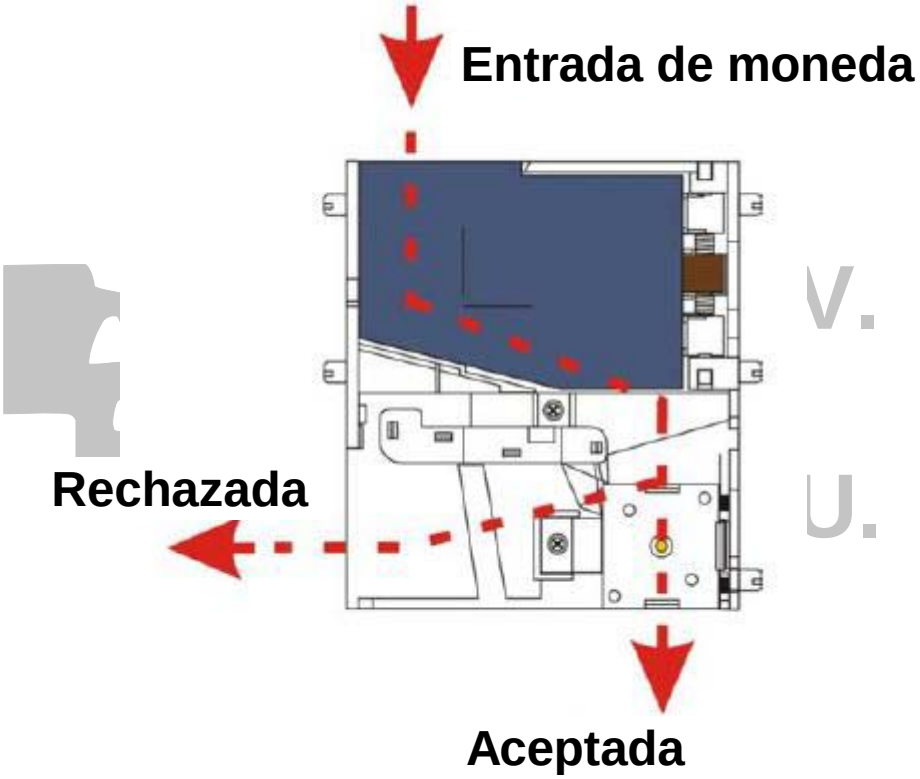

Fig. 2 Camino de la moneda en validadotes frontales

Estrategias de control del detector

La primera forma en que se puede pensar en controlar el detector se describe en el diagrama de la Fig. 3.

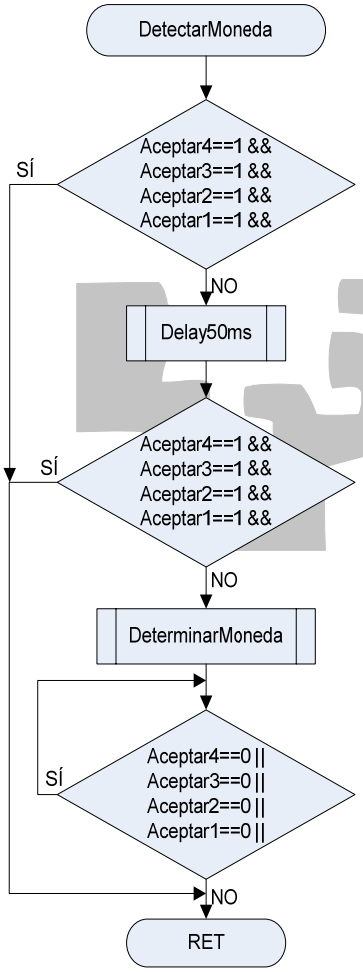

Fig. 3 Detección de billete en modo binario

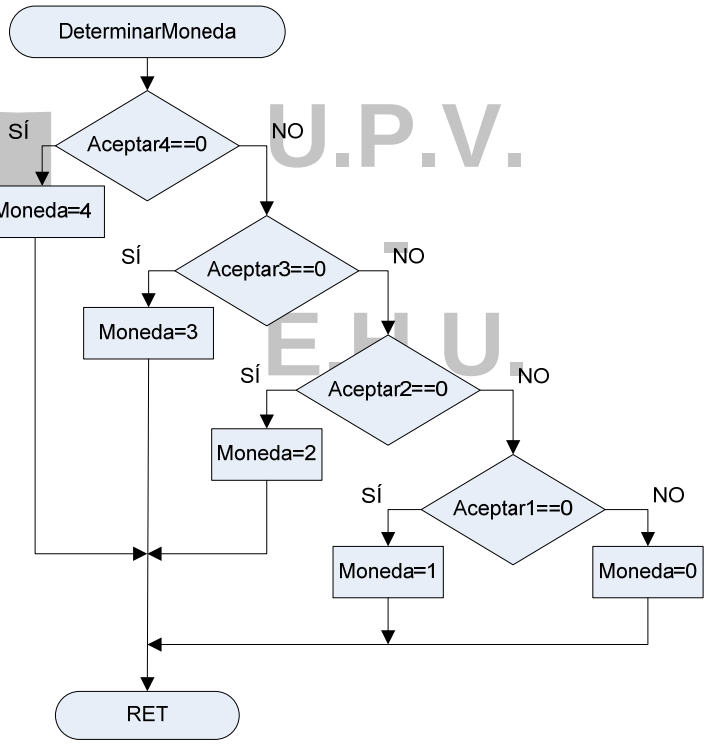

Fig. 4 Determinación del billete en modo binario

De esta forma podemos detectar las cuatro monedas de que es capaz del detector del que se dispone en el laboratorio. El hecho de comprobar el pulso a los 50ms. proviene de una recomendación del fabricante para evitar lecturas erróneas y fraudes. Así mismo, y dado de que las monedas detectables no son fracciones de euro, se ha preferido dar valores enteros que representen cada uno de los cuatro posibles tipos: 0.2€, 0.5€, 1€ y 2 €.

Si se realiza el control de esta forma, el firmware del micro estará durante unos 100ms. metido en lazos de espera que impedirían atender a otros periféricos con lo que se degradaría el funcionamiento del sistema. Para evitar esto, se puede programar de una manera distinta, usando

funciones no bloqueantes que sigan el esquema de la pseudo-máquina de estados de la Fig. 5. AceptarX respresenta a cualquiera de las líneas Accept que se ponga a nivel bajo.

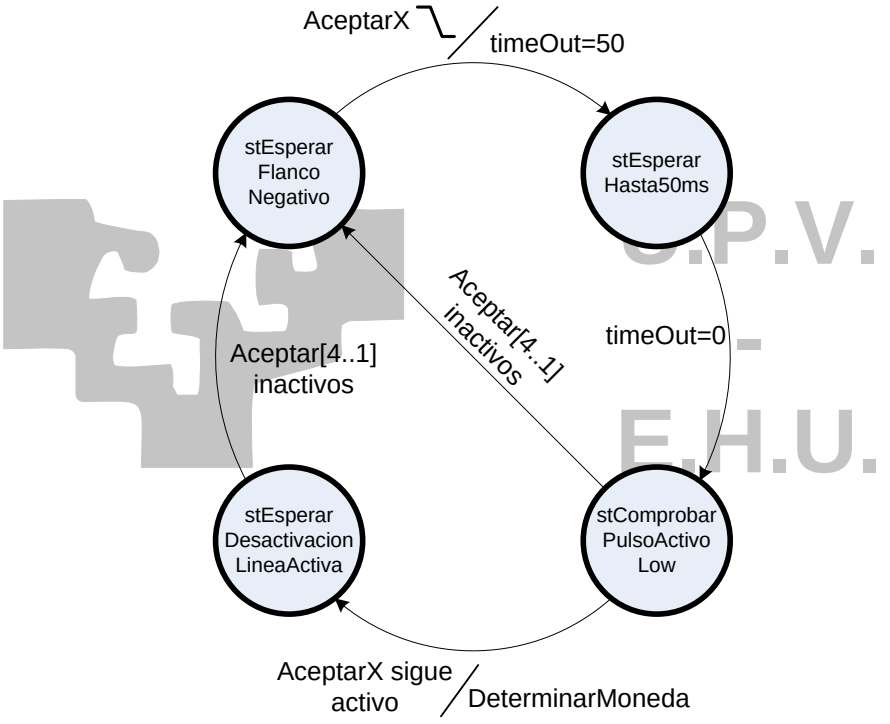

Fig. 5 Pseudo-máquina de estados para el control no bloqueante del uC

En estado de inactividad, sin haberse iniciado la detección de una moneda, el estado activo sería *stEsperarFlancoNegativo*. En el momento en que cualquiera de las líneas de aceptación se pusieran a cero, se inicializaría un timeOut de 50ms. cuya variable de conteo sería decrementeada dentro de la rutina de servicio de interrupción de alguno de los temporizadores del micro hasta llegar a cero. El valor de 50 se usa al suponerse que la ISR del temporizador se ejecuta cada milisegundo. La máquina permanece dentro de ese estado (*stEsperarHasta50ms*) hasta que se cumple el tiempo y entonces vuelve a comprobar la validez del pulso en *stComprobarPulso ActivoLow*. Si fue un pulso falso se regresa al estado inicial *stEsperarFlancoNegativo* y si no se pasa a esperar a que la línea se desactive por si misma en *stEsperarDesactivacionLineaActiva*. En ese momento se reinicia de nuevo el ciclo.

El código en C que implementaría este esquema sería similar al siguiente

```
enum stMonedero { stEsperarFlancoNegativo,
                  stEsperarHasta50ms,
                  stComprobarPulsoActivo,
                  stEsperarDesactivacionLineaActiva};

typedef enum stMonedero EstadosMonedero;
EstadosMonedero stMaqEstMonedero;

unsigned char Billete;
unsigned char volatile timeOut;

void DetectarMoneda(void){
        switch(stMaqEstMonedero){
                case stEsperarFlancoNegativo:
                        if(!Aceptar1 | !Aceptar2 | !Aceptar3 | !Aceptar4){
                                timeOut = 50;
                                stMaqEstMonedero = stEsperarHasta50ms;
                        }
                break;
                case stEsperarHasta50ms:
                        if(timeOut == 0)
                                stMaqEstMonedero = stComprobarPulsoActivo;
                break;
                case stComprobarPulsoActivo:
                        if(!Aceptar1 | !Aceptar2 | !Aceptar3 | !Aceptar4){
                                DeterminarMoneda();
                                stMaqEstMonedero = stEsperarDesactivacionLineaActiva;
                        }else
                                stMaqEstMonedero = stEsperarFlancoNegativo;
                break;
                case stEsperarDesactivacionLineaActiva:
                        if(Aceptar1 & Aceptar2 & Aceptar3 & Aceptar4)
                                stMaqEstMonedero = stEsperarFlancoNegativo;
                break;
                default:
                        //Tratamiento del error
                break;
        }
```

El objetivo principal es que cada vez que se entre dentro de la función *DetectarMoneda* no se esté más que una decena de microsegundos, lo que corresponda al *case* activo en cada momento, para poder salir y atender a otros periféricos que lo requieran.

A pesar de que la forma de implementar una estructura de control *switch-case* en ensamblador puede no ser sencilla, siempre se puede usar unos *if-then-else* anidados sobre la variable que lleva el control de estado, *stMaqEstMonedero*.

De apoyo a este esquema se necesitaría el tratamiento independiente en la ISR del un timer de la variable *timeOut* de una forma similar a la que se sugiere en el diagrama de flujo de la Fig. 6.

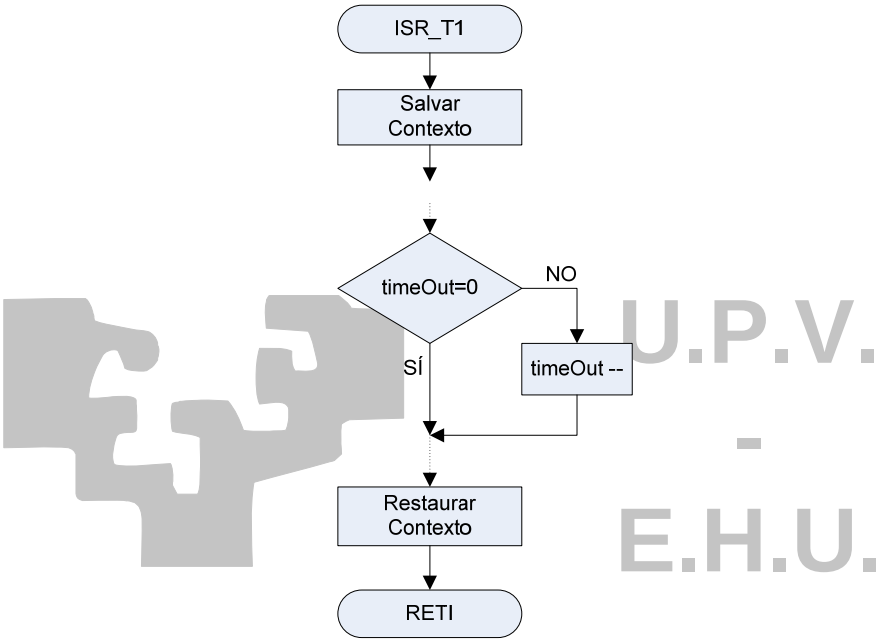

Fig. 6 Diagrama de flujo para el tratamiento del timeOut

Implementación

No se puede leer de la salida de Return ya que no está equipada la maqueta con el botón correspondiente. Los valores de las monedas y los canales de salida asociados se describen en Tabla 2.

Pin	Canal	Valor
P2.4	Aceept1	2€
P2.5	Accept2	1€
P2.6	Accept3	0.5€
P2.7	Accept4	0.2€

Tabla 2 Valores detectados por el C120

Además del detector de monedas se dispone de dos conectores molex de dos vías para la fijar sendos contadores electromecánicos que se activarán con un uno lógico cuya anchura de pulso debe ser de al menos 5ms. Durante ese tiempo el contador realiza media vuelta en su cifra de unidades. Al dejar de activar el pulso, y cesar por tanto la corriente que atraviesa la bobina, gira la

otra mitad de su recorrido. Estos contadores están unidos a los siguientes pines del 51 y se activan según el circuito descrito en la Fig. 7

Pin	Situación contador electromecánico
P3.7	Conector más cercano al de 50 vías
P3.6	Conector más alejado al de 50 vías

Tabla 3 Situación de los conectores para los contadores

Tal y como se especifica en la Tabla 1, todos los pines de los puertos P0, P1 y P2 están unidos sendos conectores que pueden albergar el lector de monedas para máxima flexibilidad a la hora de elegir el puerto con el que se quiere trabajar.

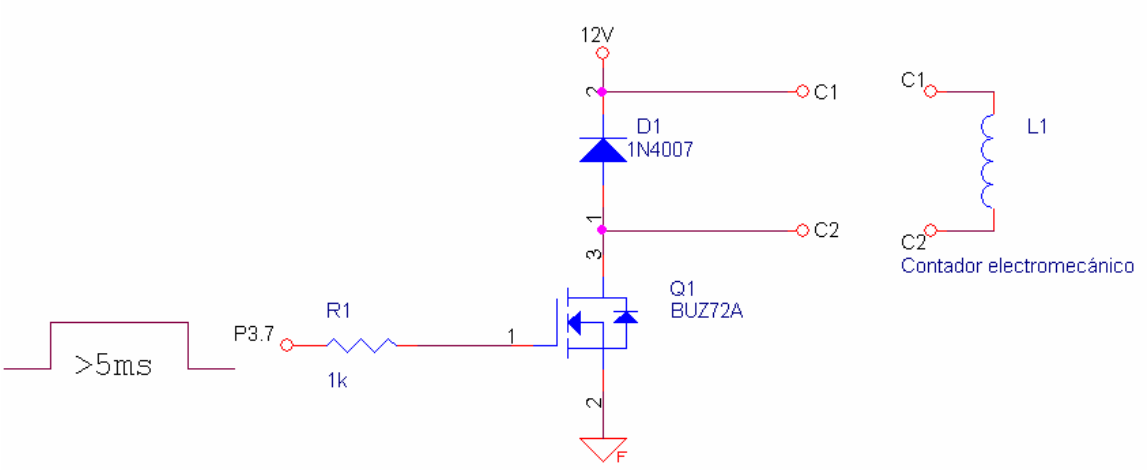

Fig. 7 Circuito para la activación de los contadores electromecánicos

Bibliografía

- C120 Technical Manual descargado de la página web www.moneycontrols.com del fabricante. Mediante registro gratuito se puede acceder a dicho manual así como a manuales de devolvedores, validadores de billetes, expendedores, etc…
- *Embedded multitasking with small MCUs: Part 1 - State Machine Constructs* Implementación de máquinas de estado para la programación de diferentes tareas. http://www.embedded.com/columns/technicalinsights/196700350?_requestid=842209
- *Embedded multitasking with small MCUs: Part 2 - Cooperative vs. pre-emptive.* Introducción a los conceptos básicos de la multitarea y a cómo usar máquinas de

estado para intentar emular capacidades similares sin sistema operativo. http://www.embedded.com/columns/technicalinsights/196701565?_requestid=841848

-

Revisiones

- 23.01.06
  - Primera versión
- 17.12.07
  - Se añaden dos artículos sobre máquinas de estados y multitarea a la bibliografía