Universidad de Huelva

Escuela Politécnica Superior
“ La Rábida “

Generador - Monitor de Vídeo para Televisión.

AUTOR: **José Juan Bautista Pulido**
TUTOR: **Andrés Roldán Aranda**

INGENIERÍA TÉCNICA INDUSTRIAL.
Especialidad: ELECTRICIDAD.
Sección: ELECTRÓNICA, REGULACIÓN Y AUTOMATISMOS.
Octubre 2002.

UNIVERSIDAD DE HUELVA.

ESCUELA POLITÉCNICA SUPERIOR
"LA RÁBIDA".

Reunido el Tribunal Examinador en el día de la fecha, constituido por:

D. .....

D. .....

D. .....

para juzgar el Proyecto Fin de Carrera titulado:

" GENERADOR – MONITOR DE VÍDEO PARA TELEVISIÓN ".

Del alumno: D. José Juan Bautista Pulido.

Dirigido por: D. Andrés Roldán Aranda.

ACORDÓ POR: _____OTORGAR LA CALIFICACIÓN DE:

_____

y, para que conste se extiende firmada por los componentes del tribunal, la presente diligencia.

La Rábida, a de de 2002.

El presidente: El secretario: El vocal:

Fdo: Fdo: Fdo:

“GENERADOR – MONITOR DE VÍDEO PARA TELEVISIÓN ”.

REALIZADO POR
José Juan Bautista Pulido.

DIRIGIDO POR
Andrés Roldán Aranda.

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA,
SISTEMAS INFORMÁTICOS Y AUTOMÁTICA.

E.P.S. "LA RÁBIDA".
UNIVERSIDAD DE HUELVA.

Huelva, Octubre de 2002.

BONARES, 15 de Octubre de 2002.

Agradecimientos.

Quiero expresar mí más sincero agradecimiento a:

-D. Andrés Roldán Aranda, tutor del proyecto, por su paciente y constante dedicación, apoyo y motivación, y por la extraordinaria influencia que, sin duda alguna, ha marcado en mí carrera. Para mí ha sido un privilegio trabajar a su lado.

-Al personal de los laboratorios de la E.P.S. "La Rábida" por haber tenido siempre tiempo para atenderme.

-También quisiera agradecer la ayuda prestada a todas aquellas personas que directa o indirectamente han contribuido en mayor o menor medida en esta labor, y en especial a aquellas personas que han confiado en mi desde el primer día y siempre han creído que lo conseguiría.

Y muy especialmente a mi familia y amigos, que en todo momento han estado a mi lado siguiendo y apoyando este trabajo.

Gracias.

ÍNDICE

Página:

CAPITULO 1: ANTECEDENTES GENERALES.

ANTECEDENTES GENERALES

1.1. COLORIMETRIA.

La finalidad última de la Televisión es presentar al espectador imágenes que le recuerden la imagen real. La percepción de estas imágenes se lleva a cabo a través de órganos sensitivos, los ojos, cuyo funcionamiento es necesario conocer someramente debido al hecho de que la televisión se adapta a las particularidades del ojo humano, aprovechándolas todas ventajosamente para el sistema de televisión.

El ojo es un mecanismo óptico complicado del que vamos a resaltar algunas propiedades.

En primer lugar es necesario recordar la naturaleza discreta de la retina. En su superficie existen cedulas sensitivas responsables de la visión y son las que transforman la radiación electromagnética incidente en la sensación visual que experimentamos.

La superficie de la retina está formada por células sensitivas de dos tipos, los conos y los cilindros. Su funcionamiento se conoce solo a través de pruebas y experimentos que sustentan algunas teorías.

En condiciones de baja iluminación sólo son suficientemente sensibles los cilindros. Los conos dejan de suministrar sensaciones. En estas condiciones sólo se percibe el brillo de los objetos y su forma, pero no los colores. Es la visión nocturna o visión escotópica. Sólo si el nivel de iluminación es suficientemente alto, los conos comienzan a suministrar sensaciones y comienzan a verse los colores. Cuando la iluminación es suficientemente alta, percibimos correctamente la forma de los objetos, su brillo y su color. Son las condiciones de visión fotópica. La situación intermedia se conoce como visión mesópica.

La situación es entonces de separación de funciones entre las células sensitivas, los cilindros que son sensibles al brillo, a la cantidad de luz, y los conos son sensibles a una cualidad de la luz, su color.

Hay otro efecto notable: el ojo se comporta como si los conos estuviesen también especializados, hay conos que presentan una sensibilidad máxima a una cierta longitud de onda.

Esto puede verificarse experimentalmente. El procedimiento consiste en iluminar la parte sensible del cono, el pigmento visual, con una fuente de luz monocroma y medir la cantidad de luz absorbida en función de la longitud de onda utilizada. De este modo se han hallado máximos de absorción en 565 nm, 530 nm y 420 nm.

Cuando incide sobre la retina una radiación no monocroma, con un espectro complejo, cada uno de estos tres tipos de cono produce una sensación que se transmite al cerebro y este la superpone dando una sensación que se interpreta como la luz original.

Este hecho da lugar a la teoría tricromática del color. Esta teoría, que usaremos profusamente en televisión, se sustenta en una amplia base experimental y explica la mayoría de los fenómenos que se observan.

La idea básica de la teoría tricromática es que es posible igualar la sensación que produce cualquier color mediante la superposición de las sensaciones que producen tres colores primarios.

A lo largo de esta introducción vamos a tratar de cuantificar esta superposición de colores y tratar de determinar bajo que condiciones, cuantitativas, se logran igualar las sensaciones. La colorimetría es la ciencia de la medición de la luz en su aspecto de color. Es una manera de "cuantificar el color" y no debemos olvidar a lo largo de la introducción que hablar de colores es hablar de sensaciones fisiológicas.

1.1.1. CURVA PATRON DE VISIBILIDAD.

Supongamos que se ilumina un ojo humano con una radiación monocroma cuya longitud de onda varia dentro del margen visible, de 380 nm (violeta) a 780 nm (rojo), manteniendo constante la potencia y pedimos a un observador medio que indique la intensidad que percibe en cada caso, normalizamos esa sensación respecto a un valor máximo y la representamos gráficamente.

Si la potencia de la fuente de luz es lo suficientemente alta, es decir el observador esta en condiciones de visión fotópica, se obtendría una gráfica parecida a la recogida en la figura 1.1 con línea continua.

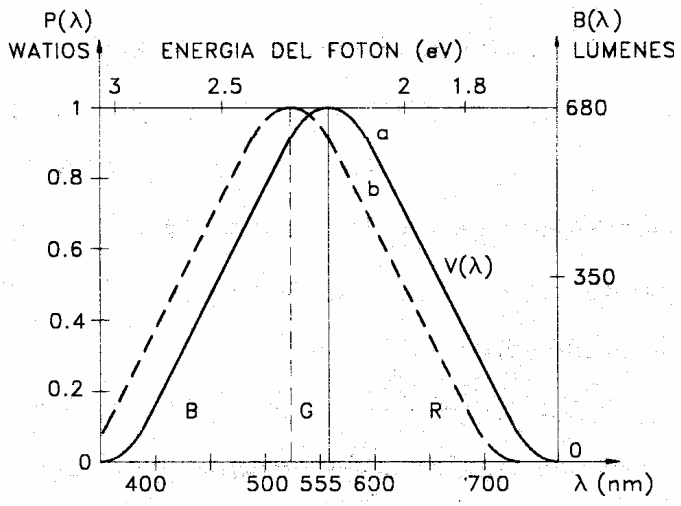


Figura 1.1 - Curva patrón de visibilidad.

Esta curva se conoce como "patrón de visibilidad fotópica" y presenta un máximo a 555 nm, produciendo dicha longitud de onda una sensación verde.

Si la iluminación no es suficientemente alta, tendremos la curva de trazos, con la misma forma que la anterior pero con el máximo ligeramente desplazado. A esta curva se la conoce como "patrón de visibilidad escotópica".

1.1.2. MEZCLA DE COLORES.

Hasta ahora hemos considerado el comportamiento del ojo cuando incide sobre él una luz simple, una radiación monocroma. A partir de ahora vamos a considerar la sensación que se produce cuando incide sobre el ojo la luz procedente de más de una fuente monocromática.

A. MEZCLA ADITIVA.

Si sobre nuestro ojo incide una determinada radiación, monocroma o no, percibimos un color. Si añadimos una o varias frecuencias más, hemos realizado una mezcla aditiva. El color percibido será distinto al que se percibiría si incidiese sólo una de las radiaciones individuales. Consideremos por ejemplo dos fuentes luminosas (roja y verde) iluminando una pared blanca; las dos radiaciones son reflejadas y la superposición de las dos fuentes da la sensación de luz amarillenta. En la zona de intersección se ha producido la mezcla aditiva de los colores rojo y verde, que produce una sensación amarilla. La mezcla aditiva de todos los colores daría el blanco.

La mezcla aditiva es un fenómeno subjetivo. Existe como sensación fisiológica.

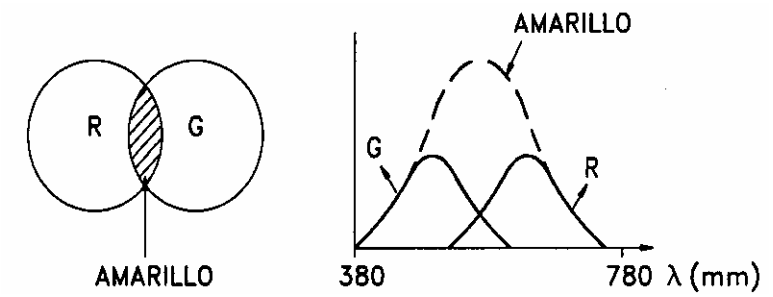


Figura 1.2 - Mezcla aditiva.

B. MEZCLA SUSTRACTIVA.

Es la complementaria de la anterior. Se trata de un efecto de filtrado, de modo que a un color determinado, se le quitan algunas componentes espectrales, resultando así otro color.

Por ejemplo, consideremos una pared blanca con dos círculos, pintados, uno rojo y otro verde. La luz blanca que se refleja es "filtrada", pues el círculo rojo absorbe todas las radiaciones menos la roja, y el verde todas menos la verde. De la intersección obtendríamos una radiación de color marrón. Es un fenómeno físico, objetivo.

La mezcla sustractiva de todos los colores daría el negro.

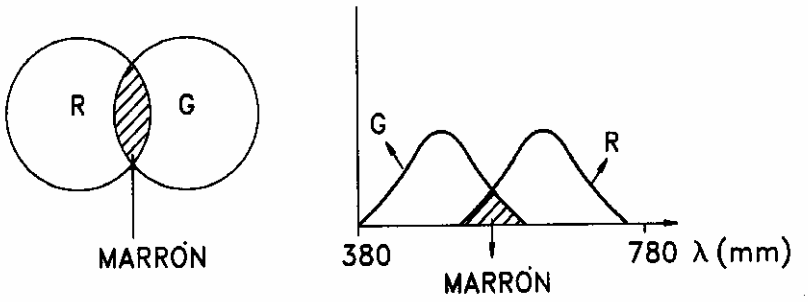


Figura 1.3 - Mezcla sustractiva.

1.1.3. LEYES DE GRASSMANN.

Son cuatro leyes básicas en las que se basa la calorimetría tricromática y por lo tanto la televisión en color. Son leyes experimentales establecidas por Grassmann.

1ª LEY DE GRASSMANN (Ley de Trivarianza).

"Los efectos que un color cualquiera (X) con luminancia Lx produce sobre el ojo, son los mismos que los producidos por la superposición de 3 colores cualquiera (A), (B), y (C), mezclados en las proporciones adecuadas La, Lb y Lc.

$$Lx\,(X) \Leftrightarrow La\,(A) + Lb\,(B) + Lc\,(C) \qquad (1)$$

Esta ley se comprueba experimentalmente mediante el colorímetro, que es un instrumento de medida en el que el observador puede evaluar la sensación de color de distintas fuentes, al observar simultáneamente dos pantallas, una iluminada con el color a igualar (X), y la otra con los tres colores de igualación.

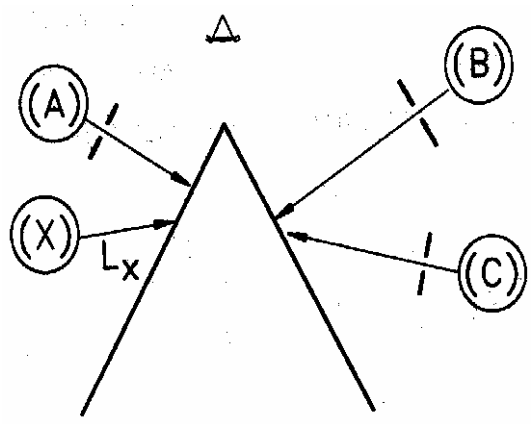


Figura 1.4 – Colorímetro.

El procedimiento de medida consiste en ajustar los irises que permiten el paso de la luz procedente de las fuentes (A), (B), y (C) hasta lograr la igualdad de sensaciones.

Por lo general, aunque la sensación visual es la misma, los espectros de ambas radiaciones son distintos. Esto significa que a una distribución espectral le corresponde una sola sensación visual, pero, a una sensación visual le pueden corresponder infinitas distribuciones espectrales. Este fenómeno recibe el nombre de "metamerismo cromático".

Puede ser necesario para igualar la sensación percibida desde cada una de las dos pantallas pasar alguno de los primarios a la otra cara del diedro. En este caso, el color en cuestión contribuiría a la ecuación (1) con luminancia negativa.

Esta situación se expresaría como:

$$Lx\,(X) + La\,(A) \Leftrightarrow Lb\,(B) + Lc\,(C) \qquad (2)$$

Una vez elegidos los 3 primarios A, B y C, las luminancias de cada uno de ellos que proporcionan la igualdad de sensación son únicas, y la sensación visual de cualquier color, puede obtenerse como combinación lineal (única) de 3 colores primarios (aunque alguna contribución sea negativa).

2ª LEY DE GRASSMANN (Ley de Luminancia).

Una vez conseguida la igualdad de sensaciones (1ª ley), la luminancia del color igualado (X) es la suma de las luminancias de los primarios utilizados.

$$Lx = La + Lb + Lc \qquad (3)$$

Esta ecuación es una igualdad numérica. La primera ley, expresada en la ecuación (1) es una igualdad de sensación visual, pero no matemática, como sí lo es la ecuación (3). Conviene además remarcar que si se cumple la ecuación (1), primera ley, se cumple la ecuación (3) pero no al revés, dos colores pueden tener la misma luminancia y ser en realidad muy distintos.

3ª LEY DE GRASSMANN (Ley de Proporcionalidad).

Conseguida la igualdad de color (1ª Ley), si se multiplican las 4 luminancias por un mismo factor, la sensación de igualdad de color en ambas caras persiste.

$$K \bullet Lx\,(X) \Leftrightarrow K \bullet La\,(A) + K \bullet Lb\,(B) + K \bullet Lc\,(C) \qquad (4)$$

Esta propiedad es muy útil si elegimos:

$$K = \frac{1}{Lx} = \frac{1}{La + Lb + Lc} \qquad (5)$$

ya que en ese caso trabajamos con colores de luminancia unidad

$$1\,(x) = \frac{La}{La + Lb + Lc} \bullet (A) + \frac{Lb}{La + Lb + Lc} \bullet (B) + \frac{Lc}{La + Lb + Lc} \bullet (C) \qquad (6)$$

4ª LEY DE GRASSMANN (Ley de Aditividad).

"Si sumando dos colores L1 (X1) y L2 (X2) obtenemos un color resultante Lx (X), podríamos haberlo obtenido sumando los primarios que igualaban cada uno de ellos".

$$L1\,(X1) \Leftrightarrow La1\,(A) + Lb1\,(B) + Lc1\,(C) \qquad (7)$$

$$L2\,(X2) \Leftrightarrow La2\,(A) + Lb2\,(B) + Lc2\,(C) \qquad (8)$$

Si se cumple:

$$L\,(X) \Leftrightarrow Ll\,(Xl) + L2\,(X2) \qquad (9)$$

Entonces:

$$L\,(X) \Leftrightarrow [La1 + La2]\,(A) + [Lb1 + Lb2]\,(B) + [Lcl + Lc2]\,(C) \qquad (10)$$

1.1.4. COMPONENTES Y COEFICIENTES TRICROMÁTICOS.

A. COMPONENTES.

Sea un color (x) con luminancia Lx. Si (A), (B) y (C) son tres colores primarios tales que:

$$Lx\,(X) \Leftrightarrow La\,(A) + Lb\,(B) + Lc\,(C) \qquad (11)$$

Entonces, las componentes del color son por definición La, Lb y Lc y, en principio, sólo pueden obtenerse de forma experimental midiendo con un colorímetro.

B. COEFICIENTES TRICROMÁTICOS.

Los coeficientes tricromáticos de un color (X) respecto a 3 primarios son, por definición, la cantidad de primarios necesarios para igualar el color (X) con luminancia unidad.

Son tres números (la, lb, lc) adimensionales, que cumplen:

$$1\,(X) \Leftrightarrow la\,(A) + lb\,(B) + lc\,(C) \qquad (12)$$

con

$$la = \frac{La}{La + Lb + Lc} \;;\; lb = \frac{Lb}{La + Lb + Lc} \;;\; lc = \frac{Lc}{La + Lb + Lc} \qquad (13)$$

Obviamente, se cumple que:

$$1 = la + lb + lc \qquad (14)$$

con lo que sólo dos de los coeficientes son independientes. De hecho, las componentes definen la luminancia y el propio color, mientras que los coeficientes sólo definen el color sin dar información sobre su luminancia.

De las componentes se pueden obtener los coeficientes, pero no al revés, a menos que se conozca la luminancia Lx. Por tanto, para realizar un sistema de televisión en color bastaría con transmitir las tres componentes de cada color, ó bien su luminancia y dos coeficientes.

1.1.5. COLORES PRIMARIOS DEL C.I.E.

El C.I.E. (Comisión Internacional de Iluminación) eligió en el año 1.931, tres colores primarios de manera que a partir de mezclas aditivas pudieran conseguirse el mayor número posible de colores con componentes positivos, ya que los componentes negativos representan un problema práctico.

Los colores elegidos fueron:

(R) Rojo espectral puro de 700 nm.
(G) Verde espectral puro de 546, 1 nm.
(B) Azul espectral puro de 435,8 nm.

Definidos estos tres primarios, cualquier color puede conseguirse como combinación de ellos, y un buen número de colores con los 3 componentes positivos.

1.1.6. TONO Y SATURACIÓN DE UN COLOR.

Para caracterizar completamente una fuente luminosa es necesario conocer sólo dos cualidades, su luminancia y su crominancia.

La luminancia es la componente Y, y se equipara subjetivamente a la claridad o luminosidad, en una escala del negro al blanco, pasando por los grises.

La crominancia es la característica cromática de un color, es decir, es la cualidad por la que distinguimos unos colores de otros, aunque tengan todos ellos la misma luminancia, y vendría dada en principio por los coeficientes del color.

La cromaticidad de un color se puede caracterizar mediante otros dos parámetros como son el tono o tinte y la saturación o pureza.

El tono, o tinte, es la longitud de onda dominante, que nos permite distinguir los colores.

La pureza o saturación de un color es el grado de mezcla con el blanco.

Los colores más puros tienen una saturación máxima, mientras que una saturación nula corresponde al blanco.

1.2. SEÑAL DE VÍDEO.

En el apartado dedicado a colorimetría se demuestra que es posible transmitir por separado la información de luminancia y dos informaciones de color, en definitiva, se establece qué información es necesario transmitir en un sistema de televisión.

En este apartado vamos a ver los procedimientos de exploración de la imagen. Fundamentalmente, se trata de determinar qué cantidad de información, que influirá directamente en el ancho de banda ocupado, tenemos que transmitir y cómo se puede organizar su transmisión para obtener una calidad de imagen que satisfaga al observador medio.

Como se verá, la señal con información útil se rodea de una gran cantidad de información en cierto modo redundante, los impulsos de sincronismo etc., cuya función veremos más adelante. En este sentido, conviene recordar el desarrollo histórico de la televisión y la obligada simplicidad circuital de los receptores primitivos.

Para seleccionar la cantidad de información a transmitir tenemos que tener en cuenta la cantidad de información que en realidad es capaz de apreciar el ojo humano.

Vamos a aprovechar dos de sus propiedades para limitar la cantidad de información a transmitir, y por lo tanto el ancho de banda necesario, a valores prácticos.

En primer lugar vamos a hacer uso de la persistencia de la imagen en la retina, la baja resolución del ojo y, en el caso de la información de color, de la mezcla aditiva espacial.

Consideremos en primer lugar el bien conocido efecto de la persistencia de la imagen en la retina. Este efecto es aprovechado en el cine, donde se presenta al ojo una secuencia de imágenes discretas que se perciben como un movimiento continuo, si el número de imágenes presentadas por unidad de tiempo es superior a un cierto umbral.

Si el numero de imágenes presentadas es inferior a dicho umbral, el ojo percibe un movimiento discontinuo parpadeante ("flicker").

En Televisión también se hace uso de este efecto presentando al observador una secuencia discreta de imágenes a una cadencia determinada, distinta de la del cine, y con algunas particularidades que se expondrán más adelante.

El segundo efecto que vamos a aprovechar en televisión es la baja resolución del ojo. En efecto, cuando un observador se sitúa a una cierta distancia de una pantalla y sobre ella se dibujan dos puntos próximos, al alejarse de la pantalla los puntos tienden a confundirse en uno solo. Este efecto es debido a la naturaleza discontinua de la retina y a la difracción que se produce al atravesar una radiación electromagnética (la luz) un iris (la pupila).El mismo efecto se obtiene si el observador se mantiene en una posición fija y los puntos se desplazan sobre la pantalla acercándose entre sí. Este efecto se puede generalizar. Imaginemos una pantalla con líneas horizontales de una determinada anchura, de modo que si el observador está cerca de la pantalla podría observarlas con nitidez. Si se aleja de la pantalla terminará viendo estas líneas como una imagen continua, perdiendo la sensación de imagen discreta que tendría un observador próximo a la pantalla.

Por último consideremos la mezcla aditiva espacial. Supongamos una situación parecida a la anterior pero ahora con tres puntos, cada uno de ellos emitiendo luz de distintos tonos y luminancias (p.e. los colores primarios). A medida que el observador se aleje de las fuentes luminosas, los tres puntos tenderán a confundirse y el ojo percibirá un solo punto cuyo color cumplirá las leyes de Grassman, es decir la sensación que le producirá será la del color resultante de la mezcla aditiva de los tres primarios cada uno de ellos con su luminancia.

En conclusión, superponiendo los tres efectos, podemos presentar al observador una secuencia de imágenes discretas, organizadas en líneas, y cada línea formada por triadas de fuentes luminosas y el observador percibirá una imagen
con movimiento continuo, sin distinguir líneas ni fuentes luminosas individuales.

1.2.1. ANALISIS SECUENCIAL DE LA IMAGEN.

De acuerdo con lo expuesto en el apartado anterior, vamos a explorar la imagen de televisión punto a punto, agrupando los puntos en líneas, como se indica en la figura 1.5.

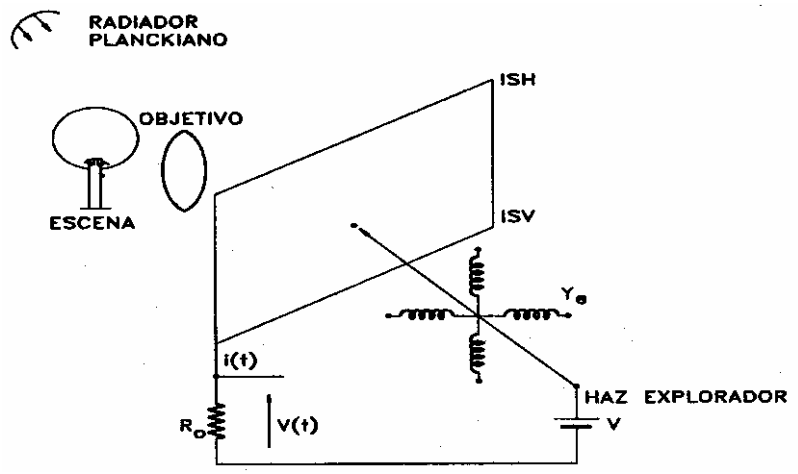


Figura 1.5 – Exploración de una escena.

En cada instante se transmite una señal eléctrica que es proporcional a la intensidad luminosa de un punto de la escena, y otra señal que llevará información del tinte y saturación del color de ese punto. En lo sucesivo llamaremos a la primera "señal de luminancia" o simplemente luminancia ya la segunda "señal de crominancia" o simplemente crominancia.

El receptor presentará esta información también punto a punto siendo esencial que la exploración de la escena sea síncrona en el transmisor y en el receptor, es decir que el receptor tenga información de cuando el transmisor ha finalizado la exploración de una línea y ha comenzado con la siguiente.

Esto obliga a introducir una información de sincronismo en la señal de TV, del modo que veremos más adelante, llamada sincronismo de línea.

La presencia de esta información indica al receptor que debe comenzar a presentar al observador la línea siguiente, de izquierda a derecha, del mismo modo que se explora en el transmisor.

Una vez explorada una imagen completa en el transmisor, es decir una vez explorado el número suficiente de líneas, debe comenzar la exploración de una nueva imagen. Al conjunto de líneas sucesivas que forman una imagen se le llama cuadro".

Cuando se finaliza la exploración de un cuadro, el receptor debe ser informado de este hecho para que comience la presentación del nuevo cuadro en el lugar adecuado de la pantalla, esquina superior izquierda. Para ello nos vemos obligados a introducir una nueva información de "sincronismo de cuadro", como veremos más adelante.

En la práctica, debemos disponer de dos transductores, uno que transforme la luz en señal eléctrica, el tubo de cámara, y otro la señal eléctrica en luz, el tubo de imagen. Ambos se basan en la exploración de una superficie, sensible o emisora según el caso, por un haz de electrones como se indica en la figura 1.5.

En el tubo de imagen, el haz de electrones tiene que ser desviado por un campo magnético creado por unas bobinas, el yugo de deflexión, recorridas por una corriente denominada corriente de deflexión con una forma en diente de sierra, de modo que en los puntos en los que la pendiente de esta corriente cambie de signo, el haz cambiará el sentido de su movimiento.

El número de electrones que transporte el haz, y la velocidad con que alcancen la pantalla, nos dará información sobre la intensidad luminosa de cada punto de la pantalla, de modo que controlando la corriente de haz, se controla la luz emitida.

1.2.2. RESOLUCION VERTICAL. NUMERO DE LÍNEAS.

El número de líneas necesarias depende de la distancia del observador a la pantalla y de su agudeza visual. Obviamente la calidad de imagen viene dada por el número de líneas, entre otros parámetros. Cuántas más líneas más calidad de imagen pero será necesario un mayor ancho de banda para transmitirlas. Es necesario llegar a un compromiso.

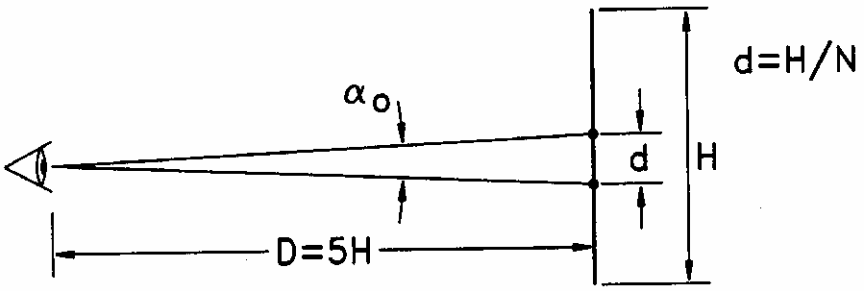


Figura 1.6 – Resolución vertical.

Considérese la figura 1.6. En ella se representa la pantalla del receptor de TV y un observador colocado a una distancia igual a 5 veces la altura de la pantalla. Sea H la altura de la pantalla y N el número de líneas.

Se llama relación de aspecto al cociente de la anchura y la altura de la pantalla.

Los sistemas actuales de Televisión tienen una relación de aspecto de 4/3, estando previsto en el futuro pasar a un valor de 16/9.

Cuando el ángulo α, bajo el cual el observador ve la pantalla, es menor que un cierto α_0, del orden de 1,5', se produce la mezcla aditiva espacial. El cerebro integra la discontinuidad de la imagen. Se tiene:

$$\mathrm{tg}\,\alpha_0 \approx \alpha_0 = (H/N) / D \qquad \Rightarrow \qquad \alpha_0 \approx 1 / 5N$$

de modo que se obtiene un número de líneas de 458. En España, se han elegido 625 líneas, de las que 575 son visibles, valor que cumple la condición anterior.

1.2.3. FRECUENCIA DE REPETICION DE CUADROS.

Una vez determinado el número de líneas de cada cuadro, es necesario determinar cuantos cuadros se van a presentar en cada segundo al observador.

El tiempo entre 2 cuadros consecutivos debe ser aproximadamente el tiempo de persistencia de la imagen en el ojo (aproximadamente 50ms).

En el cine se presentan 24 cuadros/sg, con una sensación de movimiento aceptable. En TV se han elegido 25 cuadros/sg, para poder establecer una relación con la frecuencia de red. Pero Con este valor cuando se está trazando la última línea, la primera se está empezando a borrar. Además, hay un período de oscuridad entre cada cuadro, como veremos más adelante, y todo ello da lugar a un parpadeo. Se ha observado, además, que el parpadeo es más perceptible con luminosidades altas.

Disminuir este efecto de parpadeo implicaría aumentar la frecuencia de imagen.

Aumentar la frecuencia de la imagen, o de cuadro, implica un mayor ancho de banda, ya que transmitimos mayor cantidad de información por unidad de tiempo. La solución es un artificio conocido como "exploración entrelazada".

Cada cuadro, o imagen, se divide en dos campos, el primer campo contiene sólo las líneas impares y el segundo campo sólo las líneas pares. De este modo se transmiten 50 campos por segundo, valor que coincide con la frecuencia de la red eléctrica. De este modo la cantidad de información que se transmite es la misma que antes, la contenida en los 25 cuadros, pero organizada de otro modo que evita el parpadeo.

Para simplificar el receptor y que las líneas de los dos campos queden perfectamente entrelazadas, el primer campo comienza en el ángulo superior izquierdo y termina en el centro del borde inferior de la pantalla. El segundo comienza en el centro de la parte superior de la pantalla y termina en la esquina inferior derecha. Esto obliga a utilizar un número impar de líneas. Con 625 líneas en total, resultan 312.5 líneas en cada campo.

La frecuencia de líneas, es decir el número de líneas transmitidas por unidad de tiempo es fh = 50 x 312,5 = 25 x 625 = 15.625 Hz, con lo que una línea dura un tiempo de th = 1/fh = 64 μseg. Por convenio, vamos a llamar H a la duración de una línea, 64 μseg, a lo largo de todo el texto. Se llama frecuencia de campo al número de campos por unidad de tiempo y obviamente es de 50 Hz, y un campo dura 20 mseg.

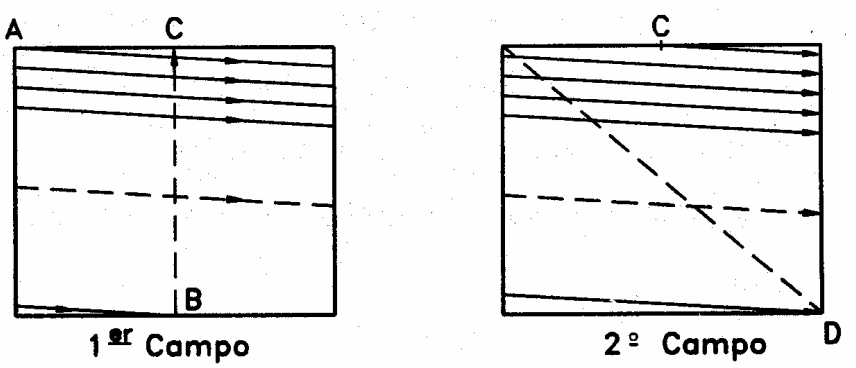


Figura 1.7 – Exploración entrelazada.

Las frecuencias de campo y línea deben ser coherentes y se generan en un circuito como el de la figura:

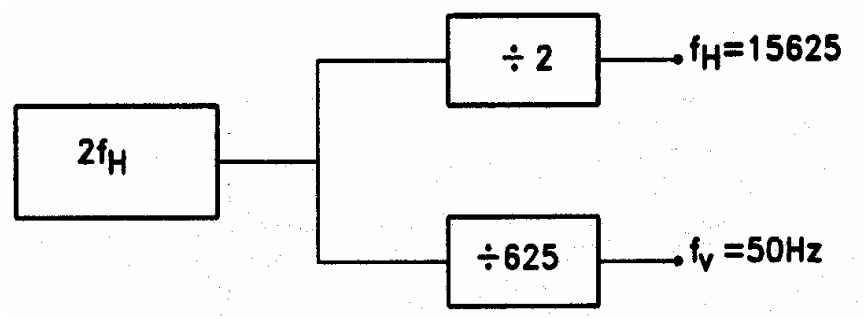


Figura 1.8 – Generación de la frecuencia de campos y la de líneas.

1.2.4. ANCHO DE BANDA DE LA SEÑAL DE VÍDEO.

En el apartado anterior, se han elegido los siguientes parámetros:
fv = 50 Hz, tv = 20 ms, fh = 15.625 Hz, th = 64 μS.

Los tiempos que dura un cuadro y una línea incluyen los de exploración propiamente dichos, barrido activo, y los tiempos de retorno del haz, o tiempos de retrazado.

Para el barrido horizontal se suelen definir los tiempos de retrazado horizontal, tfh2 y un tiempo de trazado horizontal th - tfh2 , o barrido activo. Por definición th es la duración total de la línea. Durante el barrido activo se transmite la información de video y durante el retrazado se suprime esta información dejando la señal al nivel de negro.

Estos tiempos cumplen que: tfh2 = 0,18 th.

Para el barrido vertical se define a su vez un tiempo de retrazado vertical, tfv2, y un tiempo de trazado vertical tv - tfv2, o barrido vertical activo. Aquí tv es la duración total de un campo. De modo análogo al caso horizontal, se cumple que th = 64 μseg

Para estimar el ancho de banda ocupado por la señal de video en banda de base vamos a suponer que deseamos transmitir una imagen formada por cuadros blancos y negros alternativos, como se indica en la figura 1.9, donde también se representan las corrientes de deflexión horizontal y vertical. Suponemos que la anchura del damero es exactamente 1 línea, que es el caso límite.

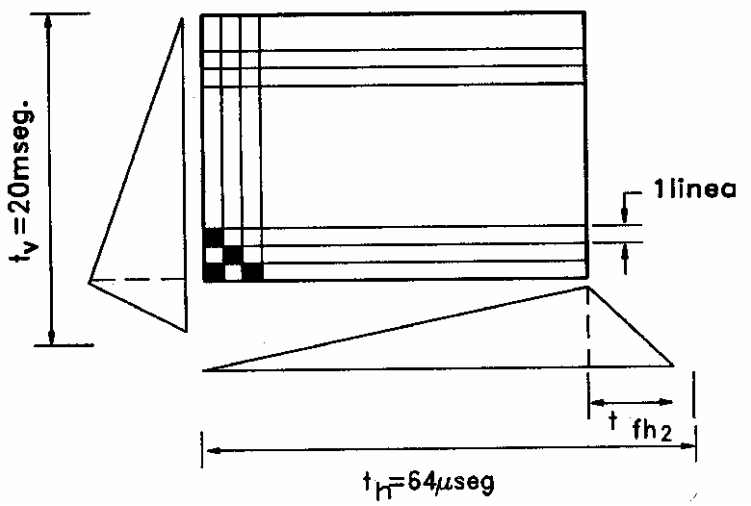


Figura 1.9 – Estimación del ancho de banda de la señal de vídeo.

-En cada columna del damero hay 625 (1- 0.08) = 575 elementos, que es igual al número de líneas visibles.

-En cada línea hay 4/3 del número de elementos de una columna, es decir 767 elementos.

En consecuencia, un cuadro tendrá 441 .025 elementos. Para barrer la parte visible del cuadro se emplea un tiempo dado por:

$$64\mu seg.\ (1-0{,}18)\ x\ 575\ =\ 30.176\mu seg.$$

con lo cual la frecuencia máxima de la señal que represente al damero vendrá dada por:

$$fr\ max.\ =\ (441025/2)\ /\ 30176\mu seg\ =\ 7{,}3MHz$$
$$=\ n^{\circ}\ de\ períodos\ /\ duración\ de\ un\ período.$$

En realidad, esta frecuencia es sólo una cota superior, en la práctica nunca hay transiciones tan bruscas en una imagen real, el haz estará ligeramente descentrado y su forma es redondeada y no cuadrada y tiene un tamaño finito, esto acarrea una perdida de definición, por ello a la frecuencia anterior se le aplica un factor de ponderación de 2/3 llamado factor de utilización. Se llega finalmente a:

$$fr = fr\ max.\ 2/3\ \approx\ 4{,}8\ MHz$$

Consideremos la situación complementaria, una imagen de luminancia uniforme. Entonces la corriente de salida tendrá un valor constante, por lo tanto el extremo inferior de la banda de base es la corriente continua. Las áreas grandes de la imagen con un contenido uniforme, darán lugar a señales de baja frecuencia, en el limite continua, y los detalles finos de la imagen darán lugar a señales de frecuencias altas, en el límite a unos 5 MHz.

1.2.5. SEÑALES AUXILIARES.

A la señal de vídeo hay que añadirle un buen número de señales auxiliares, cuya utilidad es facilitar el sincronismo del receptor y del transmisor. Se llama señal compuesta de video frecuencia (SCVF) a la señal que resulta de superponer las señales auxiliares a la señal de video. A partir de ahora, vamos a considerar una señal de Televisión normalizada en banda de base. Supongamos que la amplitud de la señal de video es 1.Se definen tres niveles importantes, representados en la figura 1.10.

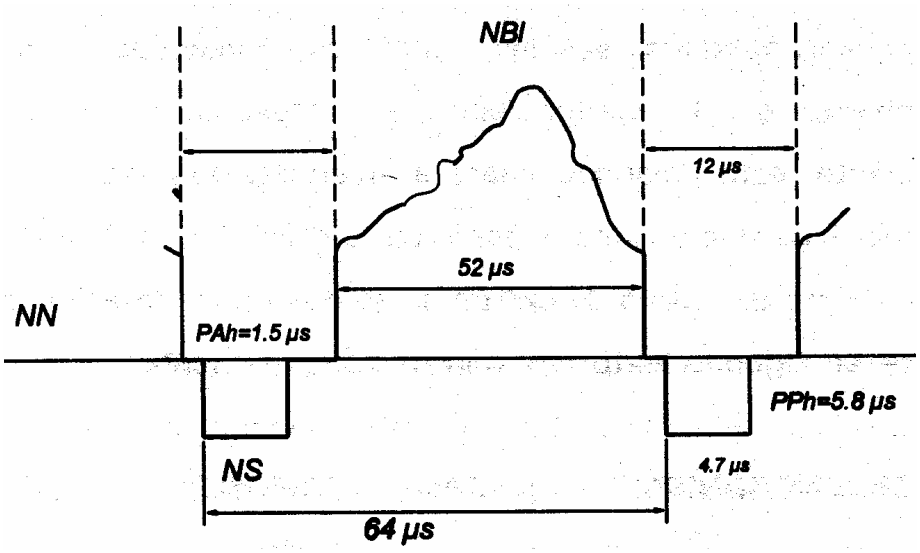


Figura 1.10 – Impulsos de sincronismo y borrado horizontal.

El nivel de negro, NN, o nivel de borrado, NB, que vamos a tomar como nivel de referencia. Cuando la señal de TV alcanza este nivel deja de verse en la pantalla imagen alguna o se ve la pantalla negra. Se le asigna una amplitud del 30% en una señal cuya amplitud total normalizada sea la unidad.

El nivel de blanco, NBI, situado al 100% de la amplitud máxima. Cuando la señal de TV alcance este valor, se emitirá luz de la máxima intensidad, dando lugar a un blanco.

El tercer nivel es el nivel de ultranegro, NUN, también llamado nivel de sincronismo o NS, con una amplitud del O % de la amplitud máxima, pero con una polaridad invertida respecto a la señal de video, de modo que si el nivel de negro provocaba una imagen negra, el nivel de ultranegro, produciría, caso de ser posible, una imagen aun con menos luminosidad.. Por supuesto, no es visible y es útil por que es a este nivel donde se fijan las amplitudes máximas de las señales auxiliares.

1.2.6. IMPULSOS DE SINCRONISMO Y BORRADO HORIZONTAL.

Los impulsos de borrado horizontal lbh se encuentran al nivel de negro (NN), o ligeramente por encima, para que no se vea el retorno de la línea, como se indica en la figura 1.10. En el impulso de borrado se distinguen:

El Pórtico Anterior (PAh) de una duración de 1,5 μs. Su utilidad es que la posición del flanco de subida del impulso de sincronismo, que determina el instante de comienzo del retroceso horizontal del haz, sea independiente nivel de luminancia del último punto de la línea anterior.

El Pórtico Posterior (PPh) de 5,8 μs de duración, sirve como nivel de referencia para fijar las amplitudes de otras señales, por ejemplo, para regular el nivel de luminancia en el receptor. Se utiliza también para enviar una señal de referencia, conocida como "burst" o salva de color, que es necesaria para recuperar en el receptor la información de color.

1.2.7. IMPULSOS DE SINCRONISMO HORIZONTAL. (ISh)

El ISh es una señal rectangular, cuyo flanco anterior marca el instante de disparo del barrido horizontal.

El ISh va desde el nivel de negro, al nivel de ultranegro. El oscilador de línea, que genera la corriente de barrido, se sincroniza con el flanco de subida. Con un ancho de banda limitado a 5 MHz, los tiempos de subida deben ser finitos, y se toman tiempos de subida y bajada de 200 a 300 ns. El instante de sincronismo, que marca el disparo del oscilador de barrido horizontal, se toma en el momento en que el flanco anterior del impulso alcanza el 50% de la amplitud. En general, todos los instantes significativos se toman en los puntos en los que la amplitud es el 50% de la máxima. Los tiempos de subida y bajada de los flancos ISh se fijan en te = 200 ns. Para los flancos del IBh, se fija te = 300 ns.

1.2.8. IMPULSOS DE SINCRONISMO Y BORRADO VERTICAL.

Los impulsos de sincronismo vertical (ISv) son anchos y están constituidos por 5 impulsos de duración 27,3 μs y separados por intervalos de 4,7 μs, llamados "serrados". Si se integran estos cinco impulsos, la señal de que se obtiene es casi idéntica a la que se obtendría al integrar un sólo impulso de duración 2.5H.

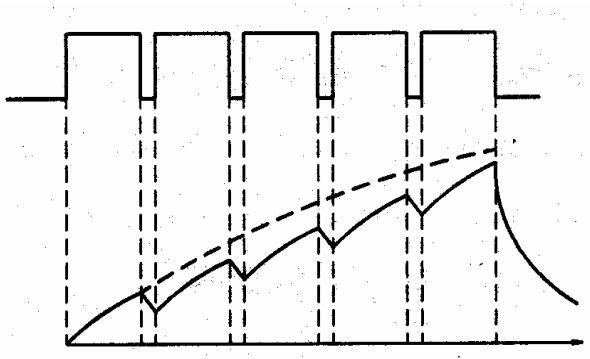


Figura 1.11 – Impulsos de sincronismo vertical integrados.

Los ISV son de mayor duración que los ISH para poderlos separar entre sí, una vez separados ambos del resto de la señal de Televisión. En efecto los ISV y los ISH, separados del resto de la señal de Televisión, dan lugar a un tren de impulsos v(t), como el representado en la figura 1.12. Haciendo pasar este tren de impulsos por un circuito integrador y otro diferenciador, seguido este último de un rectificador, se tienen las señales representadas en la figura 1.12.

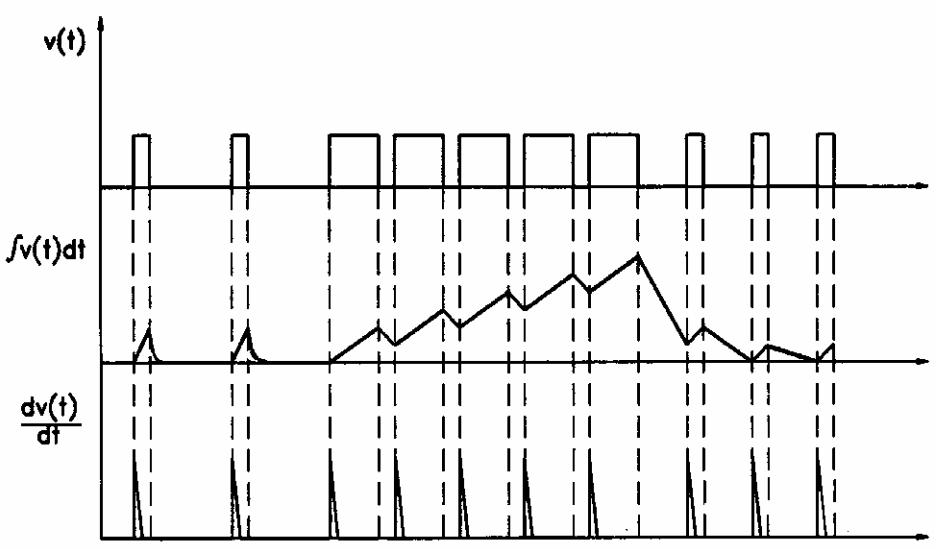


Figura 1.12 – Separación de Impulsos de sincronismo horizontal y vertical.

Los ISV, al pasar por un circuito integrador forman los impulsos de sincronismo Fin de Campo Integrado (ISVi). Este impulso es, aproximadamente, un diente de sierra exponencial.

Cuando este diente exponencial alcanza una amplitud determinada Vd, un comparador lo detecta y controla el comienzo del retrazado vertical. El final de cada campo y comienzo del siguiente viene dado por el instante en que se alcanza la tensión de control Vd.

Los flancos que se obtienen con la señal diferenciada se usan para el control del oscilador de barrido horizontal. Los circuitos de sincronismo horizontal en el receptor sólo interpretan los flancos de subida correspondientes al final de línea. Para ello se tiene un oscilador de frecuencia igual a la de línea que se engancha en fase con los flancos de subida de los ISH, pero que no responde a los flancos que aparecen en los puntos intermedios.

Las separaciones, "serrados", entre los 5 ISV permiten mantener el sincronismo horizontal durante el retorno vertical. Por esta razón, durante el ascenso vertical del haz se barren 2,5 líneas. En realidad, el número de líneas empleadas en el ascenso vertical en la práctica, depende del tiempo de retrazado, que a su vez viene determinado por el diseño de la etapa de deflexión vertical.

1.2.9. EXPLORACION DE CAMPOS.

Vamos a considerar en este apartado cómo se utilizan los distintos impulsos de sincronismo para lograr un correcto entrelazado de los dos campos.

En primer lugar considérese un caso idealizado. El primer campo comienza en la esquina superior izquierda de la pantalla (punto A) y termina en el centro de la parte inferior (punto B), en la línea 312,5. Desde aquí el haz retrocede hasta la parte central de la pantalla en la parte superior (punto C). El segundo campo comienza en el punto C, de forma que si el tiempo de retorno fuese nulo, se correspondería con la continuación de la línea 312,5 y termina en la esquina inferior derecha de la pantalla (punto D) en la línea 625. En esta figura se ha simplificado bastante el proceso real. En primer lugar hay que considerar que no todas las líneas son visibles, ni tampoco en toda su longitud, como se desprende de un análisis cuidadoso de las duraciones de los impulsos de borrado y sincronismo que se han expuesto anteriormente.

En segundo lugar durante el retrazado vertical se siguen barriendo líneas horizontalmente. Una situación más cercana a la realidad se da en la figura 1.13.

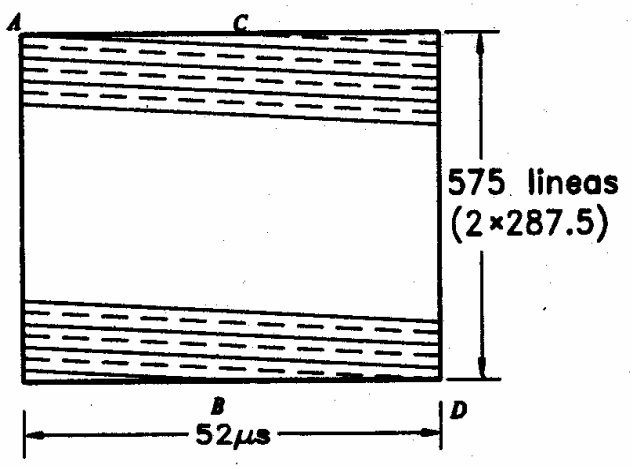


Figura 1.13 – Exploración entrelazada de una pantalla.

En ella se aprecia cómo sólo una parte del barrido es visible, con una duración de unos 52 microsegundos, y cómo en realidad la imagen que finalmente resulta visible en la pantalla contiene unas 575 líneas (2 campos de 287,5 líneas cada uno) entrelazadas. Podemos examinar con más detalle cómo se llega finalmente a esta situación, considerando la figura 1.14.

Sobre ella se ha representado una pantalla de un receptor de TV, la corriente de deflexión horizontal y los impulsos de sincronismo y borrado horizontales.

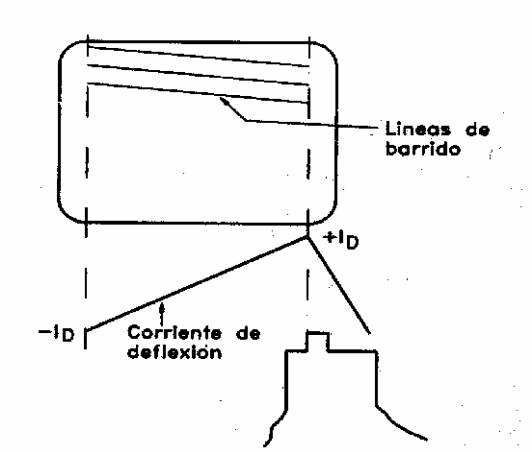


Figura 1.14 – Exploración sin sobre barridos.

Se ha supuesto que las dimensiones de la pantalla son tales que con la amplitud y la duración de la corriente de barrido, la traza sobre la pantalla del haz de electrones deflectados (RASTER) es tal que no cubre por completo la anchura de la pantalla, de modo que habrá dos franjas negras en los dos laterales de esta última.

Estas franjas negras obedecen a un doble motivo: en primer lugar porque no reciben electrones, ya que el haz no es deflectado, según nuestra hipótesis, fuera del área comprendida entre las dos líneas verticales de trazos de la figura mencionada.

Además, dentro de la zona barrida, la parte que está sobre el nivel de borrado, tampoco es visible, en este caso porque el haz de electrones estará cortado por la polarización de los electrodos de control del tubo de imagen.

Puede observarse como el flanco anterior del impulso de sincronismo horizontal provoca la inversión de la corriente de deflexión, cambiando su pendiente, de modo que el haz vuelve desde el extremo derecho de la pantalla al izquierdo (retrazado) en unos 12 μs, aunque este tiempo puede ser ligeramente menor en algunos casos..

Al invertirse las corrientes de deflexión pueden producirse en los primeros instantes oscilaciones amortiguadas que provocarían distorsiones en la imagen, para evitarlas se elige la amplitud de la corriente de deflexión de modo que exista un sobrebarrido, tal que los limites físicos de la pantalla sean ligeramente inferiores a lo que sería el raster del haz de electrones, como se indica en figura 1.15.

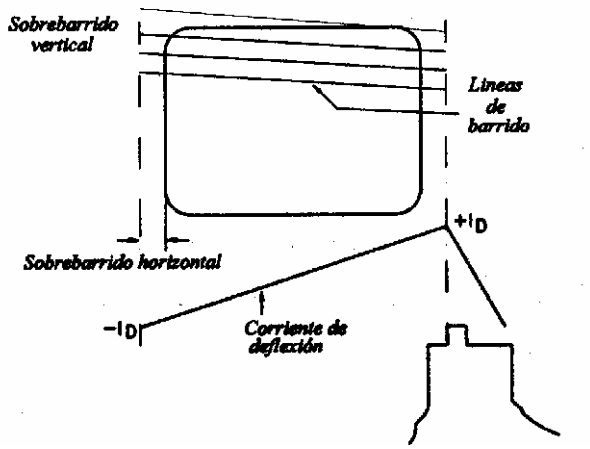


Figura 1.15 – Exploración con sobre barridos.

Para el barrido vertical, de modo análogo, también hay que considerar un sobrebarrido vertical, con las mismas funciones que el del barrido horizontal.

1.2.10. IMPULSOS DE IGUALACIÓN.

Hay que considerar un efecto adverso en el proceso de exploración que es preciso corregir. Debido a que el primer campo de cada cuadro termina en la mitad de una línea y el segundo campo del cuadro al final de una línea, al integrar los ISV en el receptor se produciría un cierto desfasaje entre el disparo del oscilador vertical en cada cuadro, ya que la tensión de control Vd se alcanza en instantes distintos (figura 1.16), con lo que cada campo tiene una duración diferente de la del anterior. En cualquier caso, la suma de las líneas de los dos campos debe ser de 625.

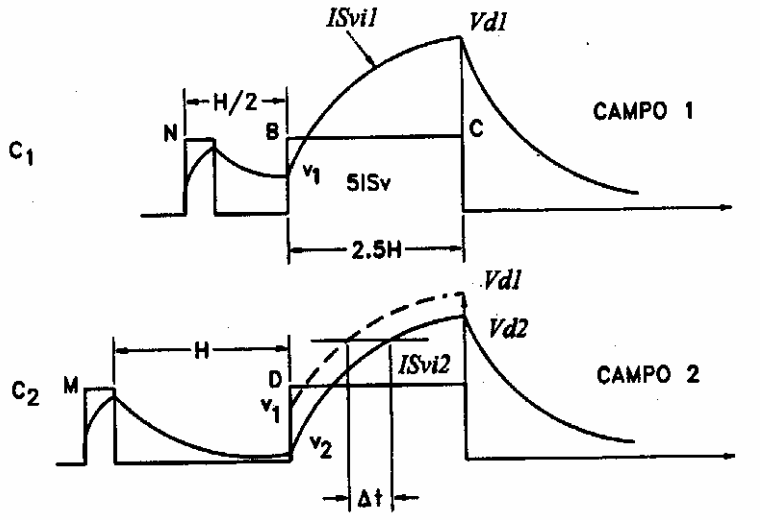


Figura 1.16 – Desigualdad de tiempos en alcanzar la tensión de disparo del oscilador vertical.

Considerando la figura 1.16, que no está dibujada a escala, se aprecia que durante el ascenso BC, Vd1 se adelanta Δt. Si Δt llegase a ser de 32 µs, aproximadamente, H/2, se podrían superponer los dos cuadros. Es preciso que el período se conserve en 20 ms exactamente, por lo que Vd1 y Vd2 deben coincidir, y por tanto V1 y V2.

Para reducir Δt a 0, se introducen, antes y después del ISV, 5 impulsos muy estrechos (2,35 μs), llamados IMPULSOS DE IGUALACION (II). La misión de los II ANTERIORES (lla) es igualar el nivel en el integrador cuando empieza el lSV, en los dos campos. Los impulsos de igualación están separados H/2, y su efecto se presenta en la figura 1.17.

Los impulsos de sincronismo vertical y los impulsos de igualación, anteriores y posteriores se colocan sobre el nivel de borrado en la SCV. En total en cada campo hay 25 líneas sin contenido de video, lo que representa unos 1.600 μs. Todo ello se muestra en la figura 1.17.

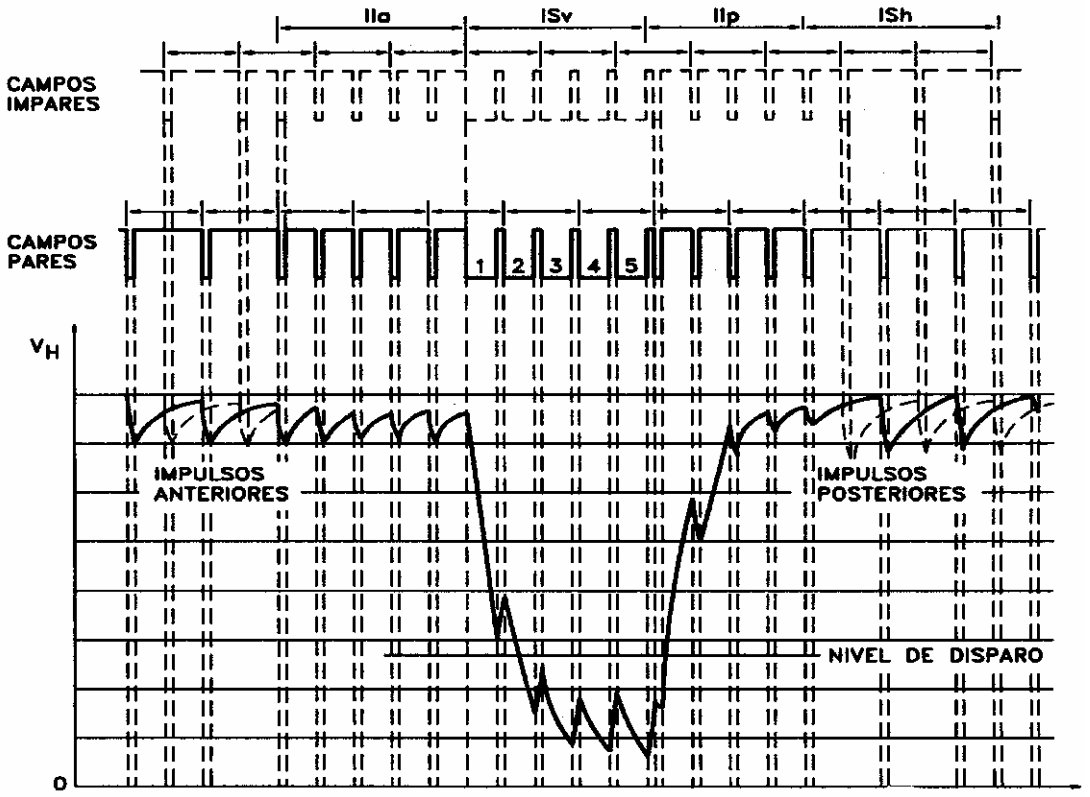


Figura 1.17 – Efecto de los impulsos de igualación.

El impulso de borrado vertical (IBV) se genera en el propio receptor de TV, y su duración no tiene que con coincidir con las 25 líneas antes mencionadas, de hecho lo habitual es que dure unas 22 ó 23 líneas. El retrazado comienza en el flanco de subida del impulso de borrado vertical y dura aproximadamente 11 líneas.

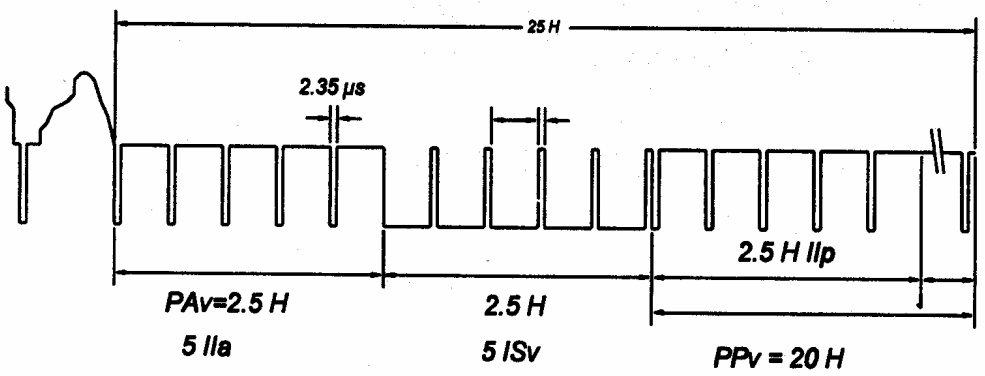


Figura 1.18 – Impulsos de igualación, anteriores y posteriores, e impulsos de sincronismo y borrado vertical.

El motivo de la inserción del IBV es evitar que pudieran llegar a verse en la pantalla las líneas barridas durante el retroceso vertical del haz. Su presencia también impide que las oscilaciones y alinealidades en el barrido, provocadas por la inversión brusca de la corriente de deflexión en el yugo de deflexión vertical, provoquen efectos desagradables en la imagen observada.

Teniendo en cuenta que en cada campo hay 25 líneas sin información de imagen, finalmente se dispone de 575 líneas visibles y 50 no visibles, en total 625.

En algunas de las líneas suprimidas pueden transmitirse señales adicionales, por ejemplo el Teletexto.

Finalmente puede considerarse la figura 1.19. En ella se representa la disposición de los impulsos que configuran dos campos consecutivos de un sistema de exploración entrelazada descritos en los apartados anteriores.

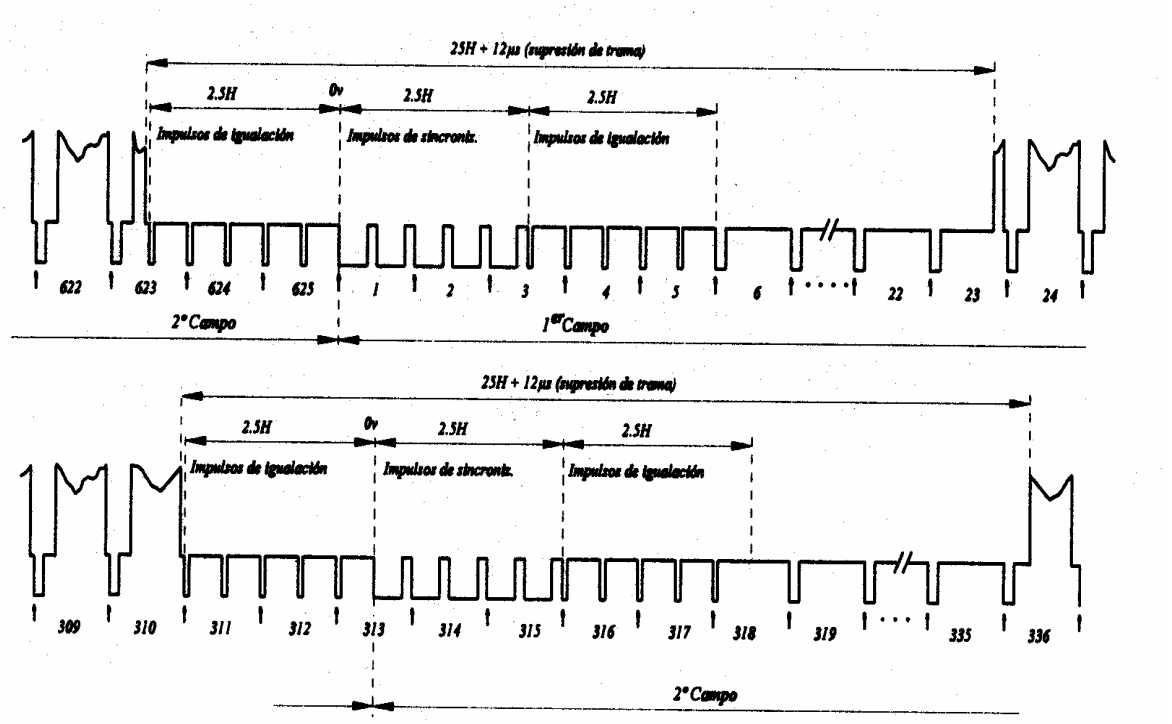


Figura 1.19 – Señales de sincronismo para la exploración de un cuadro completo.

La primera línea del primer campo comienza en el flanco rotulado como OV sobre la figura. A continuación se barren líneas en sincronismo con los flancos marcados con una flecha sobre la figura, generándose una rampa de corriente de barrido horizontal.

Las líneas 1 a 23 no se ven por un doble motivo: en primer lugar, la información contenida en ellas se encuentra al nivel de negro y en segundo lugar, si no fuese así, la presencia del impulso de borrado vertical impediría su visión.

Desde la 23 son todas visibles hasta el final de la línea 310, a partir de la cual y hasta la línea 335.5, la información contenida está al nivel de negro. En la línea 312.5 comienza el segundo campo. Desde la línea 335,5 hasta la 622,5 son todas visibles. Desde la línea 622.5, hasta la 625 son líneas no visibles, al estar al nivel de negro.

1.3. INFORMACIÓN DE COLOR.

Históricamente la Televisión se desarrolló comenzando por los sistemas monocromos o en blanco y negro. Eran sistemas simples, a los que fue necesario añadir la información de color a posteriori.

Este hecho obligó a desarrollar procedimientos de codificación de la información de color que permitieran la compatibilidad de los sistemas monocromos en explotación con los sistemas en color que se pretendía implementar, lo que condicionó enormemente su desarrollo y sus características.

Fue necesario respetar los planes de frecuencias, la canalización y muy especialmente conservar la compatibilidad directa e inversa entre el sistema en color y los sistemas monocromos preexistentes.

Por compatibilidad directa se entiende que una imagen con información de color pueda ser recibida por un receptor monocromo y por compatibilidad inversa que un receptor para señales con información de color pueda recibir una transmisión monocroma.

Todo ello configuró condicionantes muy significativos para elegir los procedimientos de codificación de la información de color.

El primer sistema en desarrollarse fue el sistema NTSC. De él deriva el sistema PAL. El tercer sistema en explotación actualmente se basa en un principio completamente diferente al de los anteriores, el sistema SECAM.

1.3.1. COMPATIBILIDAD Y PRINCIPIO DE LUMINANCIA CONSTANTE.

Para representar toda la gama de colores posibles con los tres primarios elegidos podríamos optar por un sistema con 3 señales en paralelo, una para cada componente R, G, B.

Este sistema de televisión en color, concebido como la simple superposición de tres sistemas monocromos, sería incompatible con el sistema de televisión monocromo y ocuparía 3 x 5 MHz de ancho de banda.

Para conseguir la compatibilidad el primer paso es generar, a partir de los tres primarios R, G, y B, una señal de luminancia idéntica a la usada en la televisión monocroma, dada por la ecuación:

$$Y = O.3R + O.59G + O.11B \qquad (1)$$

La información de color, que en lo sucesivo se designará como información de crominancia, se genera también a partir de las componentes R, G y B como se explica en los apartados siguientes, y se transmite superpuesta a la señal de luminancia de modo que un receptor monocromo sólo aproveche la señal de luminancia sin verse perturbado por la señal de crominancia. Un receptor en color en cambio usará ambas, crominancia y luminancia, pero si en una transmisión dada sólo existiera información de luminancia podría presentar una imagen monocroma sin dificultades.

PRINCIPIO DE LUMINANCIA CONSTANTE.

En la discusión anterior, se ha supuesto implícitamente que la luminancia de cada elemento de imagen es independiente de su contenido cromático, y por lo tanto independiente de que se transmita en blanco y negro, o en color. La televisión en color se basa en esta hipótesis conocida como "principio de luminancia constante", que establece que la señal de luminancia contiene información únicamente del brillo de las escenas, y que las señales de crominancia no contribuyen a aquella.

Si se cumple lo anterior, las señales interferentes en el canal de cromaticidad no deberían producir efectos sobre la luminancia en el receptor.

Color:	R	G	B	Y
Blanco	1	1	1	1.00
Amarillo	1	1	0	0.89
Cian	0	1	1	0.70
Verde	0	1	0	0.59
Magenta	1	0	1	0.41
Rojo	1	0	0	0.30
Azul	0	0	1	0.11
Negro	0	0	0	0.00

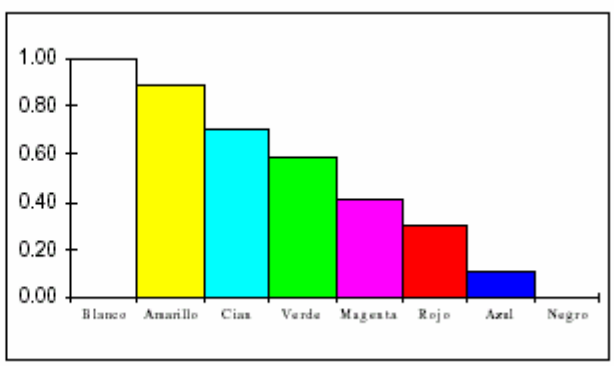


Figura 1.20 – Representación del brillo relativo de colores primarios y secundarios.

1.3.2. SEÑALES DIFERENCIA DE COLOR.

Según se ha visto en el apartado de colorimetría, para representar un color necesitamos dos señales de información de color, además de la propia luminancia. Se ha elegido enviar dos de las tres señales de "diferencia de color", definidas por las ecuaciones siguientes:

$$CdR = R - Y \; ; \; CdB = B - Y \; ; \; CdG = G - Y \qquad (2)$$

Estas tres señales tienen la propiedad de anularse si las componentes R, G y B cumplen la condición R = G = B, es decir, son nulas en los blancos, grises y negros.

$$CdR = 0 \; ; \qquad CdB = 0 \; ; \qquad CdG = O \qquad (3)$$

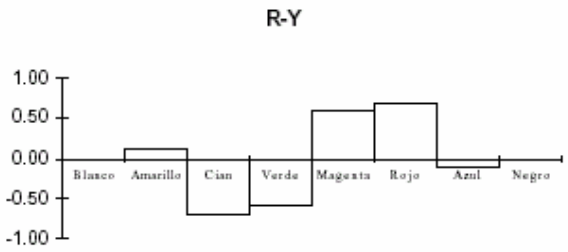
Esta propiedad es útil, y, de hecho, motivó la elección de las señales de diferencia de color, porque si se desea transmitir sólo información de brillo, o de luminancia, estas tres señales se anulan con lo cual se facilita la compatibilidad con los sistemas monocromos preexistentes.

De las tres señales de diferencia de color, sólo dos son necesarias en realidad. Estadísticamente se comprobó que la señal de diferencia de color verde conducía en la mayor parte de los casos a relaciones señal/ruido menores que las otras dos, por lo que se eligió finalmente transmitir las señales de diferencia de color roja y azul.

$$CdR = R - Y = O.\ 7R - 0.59G - 0.11B \qquad (4)$$

$$CdB = B - Y = -0.3R - 0.59G + 0.89B$$

Color:	R	G	B	R-Y
Blanco	1	1	1	0.00
Amarillo	1	1	0	0.11
Cian	0	1	1	-0.70
Verde	0	1	0	-0.59
Magenta	1	0	1	0.59
Rojo	1	0	0	0.70
Azul	0	0	1	-0.11
Negro	0	0	0	0.00



Color:	R	G	B	B-Y
Blanco	1	1	1	0.00
Amarillo	1	1	0	-0.89
Cian	0	1	1	0.30
Verde	0	1	0	-0.59
Magenta	1	0	1	0.59
Rojo	1	0	0	-0.30
Azul	0	0	1	0.89
Negro	0	0	0	0.00

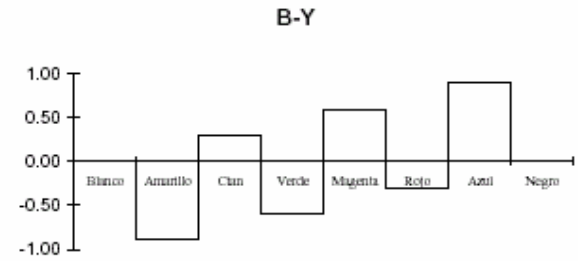


Figura 1.21 – Forma de las señales diferencia de color.

1.3.3. ANCHO DE BANDA DE LA SEÑAL DE CROMINANCIA.

Una vez elegidas qué señales se van a enviar es necesario averiguar qué ancho de banda es necesario utilizar en cada una de ellas. Nuevamente, para elegir este importante parámetro hay que considerar como funciona el ojo humano. En efecto, el ojo tiene menos cantidad de conos (células sensibles al color) que de cilindros {células sensibles a la luminosidad).

Esto provoca que la agudeza del ojo para distinguir dos detalles coloreados con la misma luminancia sea cinco veces menor que para distinguir dos detalles con distinta luminancia de igual ó distinta crominancia.

Esta conclusión es experimental, y permite reducir el ancho de banda de las señales de crominancia (diferencia de color) a 1/5 del ancho de banda para la señal de luminancia.

En la norma europea, con un ancho de banda de luminancia de 4.8MHz, la señal de crominancia se transmite con un ancho de banda de aproximadamente 0.9MHz.

1.3.4. MODULACIÓN EN CUADRATURA DE LAS SEÑALES DIFERENCIA DE COLOR.

Las señales de diferencia de color (R - Y), (B - Y) modulan en amplitud, con portadora suprimida, a dos portadoras de igual frecuencia, fsc, y en cuadratura, es decir con un desfasaje de 90 grados.

La señal de crominancia vendrá dada en principio por C1:

$$C1 = (B - Y) \bullet \operatorname{sen} \omega_{sc} t + (R - Y) \bullet \cos \omega_{sc} t \qquad (5)$$

esta señal puede considerarse como la superposición de dos fasores, R1 y R2 dados por:

$$R1 = (B - Y) \bullet \operatorname{sen} \omega_{sc} t \qquad (6)$$

$$R2 = (R - Y) \bullet \cos \omega_{sc} t$$

La resultante es un fasor $|R| \cdot e^{j\phi}$ donde

$$|R| = \sqrt{|R1|^2 + |R2|^2} \qquad (7)$$

$$\phi = \text{tg-1} \frac{|R2|}{|R1|} \qquad (8)$$

En nuestro caso será:

$$|C1| = \sqrt{(B-Y)^2 + (R-Y)^2} \qquad (9)$$

$$\phi 1 \quad = \text{tg-1} \frac{(R-Y)}{(B-Y)} \qquad (10)$$

Pasando al dominio del tiempo, podríamos escribir C1 de la forma:

$$C1 = \sqrt{(B-Y)^2 + (R-Y)^2} \bullet \operatorname{sen}(\omega_{sc} t + \phi 1) \qquad (11)$$

Escrito de esta forma pueden observarse tres propiedades fundamentales:

- La portadora queda modulada simultáneamente en amplitud y fase.
- El ángulo $\phi 1$ representa la orientación del fasor C1 en el plano de diferencias de color, y por tanto contiene información sobre el tinte o tono del color.
- El módulo del fasor es la distancia entre el blanco y el color que se desea transmitir, por lo tanto el módulo de la señal de crominancia lleva información sobre la saturación del color.

Como se ha indicado antes, la señal de crominancia C1 se superpone con la señal de luminancia Y para obtener la señal de vídeo de televisión en color, es decir se suman en el dominio del tiempo las señales Y y C1.

Esta superposición obliga a introducir unas modificaciones sobre la señal de crominancia:

-Ya que la señal (B-Y) toma normalmente valores mucho mayores que la (R-Y), esta multiplica por el factor, elegido empíricamente, 0.56.

-Si se suman directamente luminancia y crominancia, para algunos colores (muy saturados), se podría producir una sobremodulación que exceda los niveles de blanco y de negro hasta un 40%. Por ello se reduce la señal de crominancia en un factor de 0.88.

Si dibujamos el vector de crominancia en coordenadas polares para los seis colores de la carta de barras tenemos una representación como la de la figura 1.22.

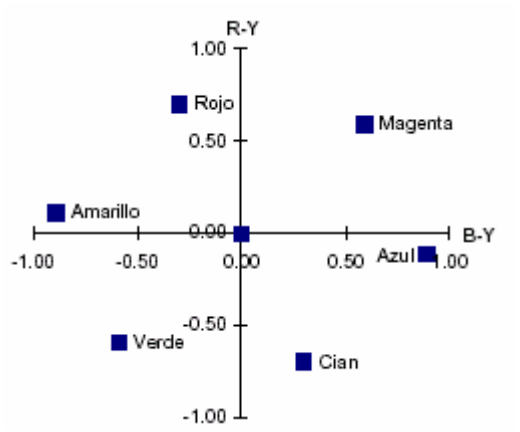


Figura 1.22 - Existe un instrumento de medida que realiza la representación polar de la señal de crominancia conocido como vectorscopio.

Para recuperar las señales de color separadas en la recepción, se precisa una detección síncrona de cada una de las componentes en cuadratura. Necesitamos un oscilador enganchado en frecuencia y fase (sincronismo) con la subportadora de color.

Para conseguir este sincronismo, se transmite una muestra de la subportadora de color denominada SALVA o BURST, situada en el pórtico posterior del IBh.

ELECCIÓN DE LA FRECUENCIA DE LA SUBPORTADORA (PAL).

Para intercalar los espectros de luminancia y crominancia PAL es necesario que ninguna línea espectral de la subportadora modulada coincida con líneas espectrales de la luminancia. Observando el espectro de luminancia, se ha de elegir:

$$fsc = (2m + 1) \cdot \frac{fh}{4}$$

y eligiendo $(2m + 1) = 1135$

$$fsc = 1135 \cdot \frac{15625}{4} = 4.4336MHz$$

siendo posible de este modo la imbricación de los espectros.

1.3.5. EL SISTEMA PAL (Phase Alternation Line).

En el procesado y transmisión de la señal de televisión desde el estudio hasta el receptor, puede darse el caso de que la señal de crominancia sufra cambios indeseados de fase con respecto a la fase de la salva, por ejemplo, debido a no linealidades en los circuitos, la fase de la señal de crominancia puede depender de la amplitud de la luminancia. A esta intermodulación se la conoce como fase diferencial. A pesar de que determinadas perturbaciones se intentan corregir en las transmisiones, no es posible en general suprimirlas por completo.

Independientemente de su origen los cambios en la fase de la señal de crominancia en el sistema NTSC representan cambios en el tono del color, lo que altera el contenido cromático de la imagen.

Bajo estas alteraciones del tono de los colores, la calidad de la imagen no es satisfactoria, y los receptores incorporan un mando, que debe ajustar el observador en base a criterios subjetivos, por ejemplo el color de la piel humana, para lograr una reproducción correcta del color. Este mando ajusta la fase de la subportadora que se recupera en el receptor hasta en 45 grados, de modo que la reproducción sea aceptable.

En los sistemas NTSC modernos, el ajuste de tono en le receptor se realiza de forma automática, por medio de señales de referencia que envía la emisora en líneas no visibles, concretamente sobre el impulso de borrado vertical. Estas señales de referencia se denominan señales VIR (Vertical Interval Reference).

Las señales VIR se empezaron a transmitir a finales de los 70. Los receptores modernos incorporan la circuiteria que decodifica las señales VIR y corrigen el tono de forma automática.

SISTEMA PAL.

El sistema PAL se basa en la inversión en líneas alternas de la señal R-Y, para corregir los errores de color propios del sistema NTSC. Supongamos que la información de crominancia es prácticamente idéntica en 2 líneas consecutivas de un mismo campo. Si el vector de crominancia es $C\varphi$ y aparece un error de fase α, los vectores recibidos en 2 líneas consecutivas serán $C\varphi+\alpha$ y $C-\varphi+\alpha$.

En el receptor (decodificador PAL) se vuelve a invertir la señal R-Y en líneas alternas, cambiándose el signo de la fase del vector de crominancia.

El ojo realiza la integración espacial de los colores de las dos líneas consecutivas en un mismo campo, y verá un color intermedio, que es el correspondiente a la suma de los vectores Cφ+α y Cφ-α que tiene la misma fase (tono) que el correcto Cφ.

La señal de crominancia, en el sistema PAL, se escribe:

$$C(t) = O.49(B\text{-}Y)\operatorname{sen}\omega_{sc} t \pm O.88(R\text{-}Y)\cos\omega_{sc} t =$$

$$= \sqrt{U^2 + V^2} \bullet \operatorname{sen}(\omega_{sc} t + \theta)$$

donde U y V son las señales B-Y y R-Y ponderadas por los coeficientes 0.49 y 0.88 respectivamente.

Para demodular esta señal de fase alternada, hace falta conocer en cada línea el signo de R-Y. Esto se consigue transmitiendo en un cierto instante de tiempo una muestra de la subportadora, la salva, con fase de ±135 grados alternativamente (dicho signo coincide con el de la señal de crominancia C(t)).

El sistema PAL corrige el error en el tono de los colores debido a cambios de fase de la crominancia, como hemos visto, pero en cambio da lugar a un error de saturación. Es decir, el error de fase se transforma en un error en la saturación del color, menos molesto que el error de tono que produciría un error de fase, pero que obliga a que los receptores PAL tengan mando de saturación, aunque no de tono.

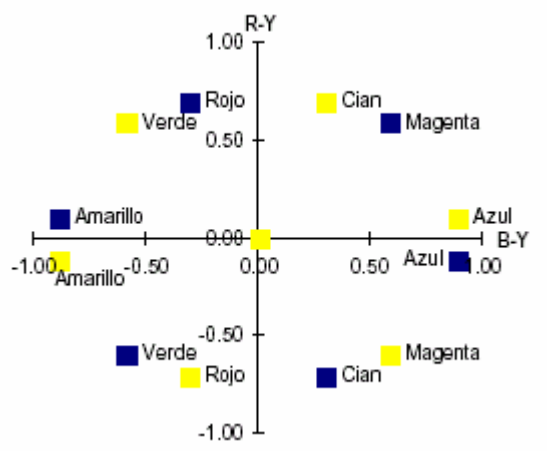


Figura 1.23 – Señal de crominancia PAL en vectorscopio, donde puede observarse la alternancia de fase.

CAPITULO 2: OBJETIVO DEL PROYECTO.

OBJETIVO DEL PROYECTO

El objetivo del proyecto es el estudio, diseño, y construcción de un Generador – Monitor de Vídeo para Televisión.

El destino del equipo será el laboratorio de televisión de la E.P.S. La Rábida, y su finalidad se extiende desde el campo de la enseñanza de la señal de video, hasta la reparación y ajuste de equipos receptores de televisión.

El equipo realizará dos funciones totalmente independientes:

1ª Función: Generador de Vídeo.

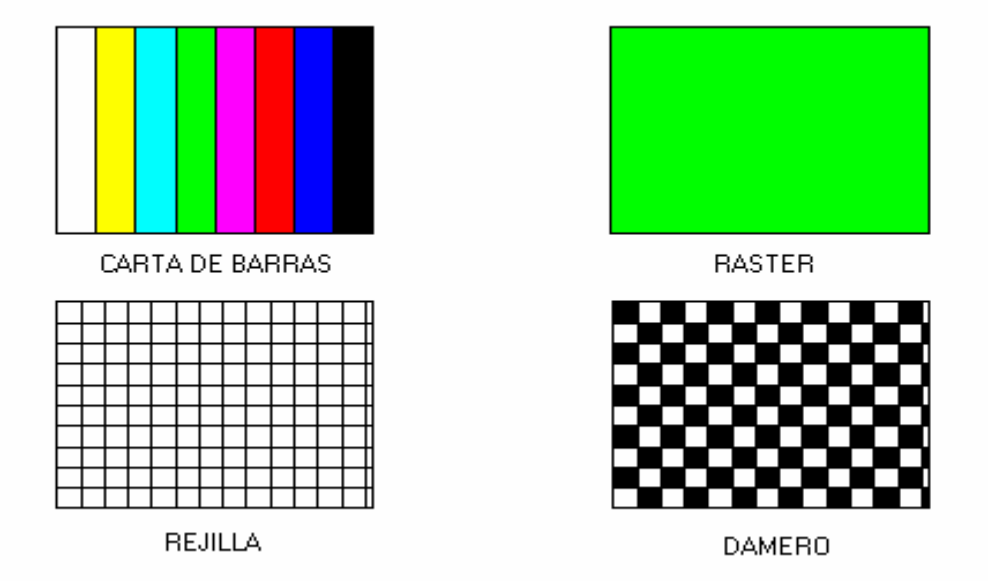


Figura 2.1 – Patrones de video más frecuentes.

El generador de vídeo a de poseer las siguientes funciones:

1) Generar varios Patrones de Vídeo a Color según las normas de nuestro país (Normas B y G).
2) Disponer de Controles Adicionales que actúen sobre las siguientes señales: Componentes de Color, Luminancia, Crominancia, Burst, y Nivel de Salida.
3) Poseer Salida de Vídeo-Compuesto, y Salida de Súper-Vídeo.

2ª Función: Monitor de Vídeo.

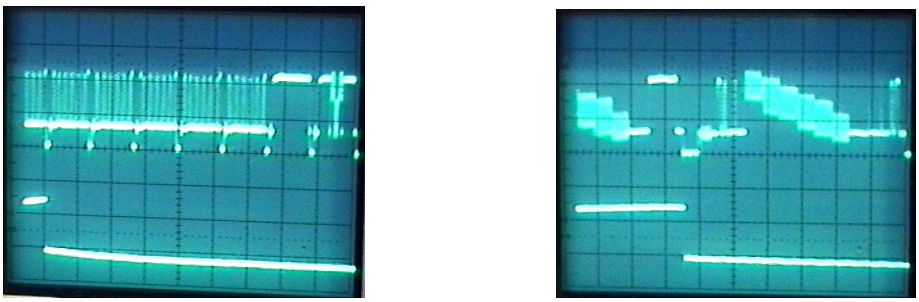


Figura 2.2 – Líneas de vídeo en osciloscopio.

El monitor de vídeo a de poseer las siguientes funciones:

1) Visualizar las Líneas de Vídeo en diferentes grupos según la finalidad de estas: Líneas de Vídeo, de Teletexto, y de Control de Calidad (V.I.T.).
2) Disponer de Teclado y Display, para la selección de los diferentes modos de funcionamiento, así como para mostrar la información que corresponda. También habrá un Buzzer que indique los sucesos importantes del sistema.
3) Disponer de una Entrada por donde introducir la Señal de Vídeo objeto de estudio, y una Salida que muestre la Señal Seleccionada, donde se conectará el osciloscopio.

Se tienen unas restricciones iniciales que son:

1) Aunque el generador y el monitor son realmente equipos independientes y pueden funcionar de forma aislada el uno de otro, los dos se montarán en la misma placa de circuito impreso.
2) Habrá una única fuente de alimentación para los dos equipos, y esta también irá alojada en la placa de circuito impreso.
3) Se utilizará un display de cuatro líneas proporcionado por el director del proyecto D. Andrés Roldán Aranda.

También se tiene como objetivo la construcción de dos generadores de video, pertenecientes a aficionados a la televisión, los cuales se encuentran disponibles en la red(Internet). Toda la información disponible para la construcción de dichos generadores se describe en el siguiente capítulo.

CAPITULO 3: DESCRIPCIÓN GENERAL DEL SISTEMA.

DESCRIPCIÓN GENERAL DEL SISTEMA

En este capítulo se realizará una breve descripción de las partes que constituyen el sistema, dejando la memoria y los cálculos para el siguiente capitulo. Para ello se darán unos diagramas de bloques, junto con la enumeración de los componentes de los mismos.

Debido a que el equipo realiza dos funciones totalmente independientes, de ahora en adelante los describiremos y analizaremos por separado (exceptuando la fuente de alimentación que es común), y nos referiremos al Generador de Vídeo como Generador, y al Monitor de Vídeo simplemente como Monitor.

3.1. DESCRIPCIÓN DEL GENERADOR.

En el diagrama de bloques de la figura 3.1 se muestran las diferentes partes que constituyen el Generador, donde se pueden observar los siguientes módulos:

- **Fuente de Alimentación:** Basada en reguladores de tensión integrados de la serie 78XX. Suministra tensiones de 5 y 12 voltios necesarias para la alimentación de los circuitos integrados.

- **Microcontrolador:** El Microcontrolador es un PIC16F84-10 de Microchip. Recibe información del teclado que le informa si debe cambiar el patrón de vídeo. Una vez seleccionado el patrón de vídeo, el microcontrolador genera las componentes de color y la señal de sincronismo compuesto, ambas necesarias para la generación del vídeo compuesto.

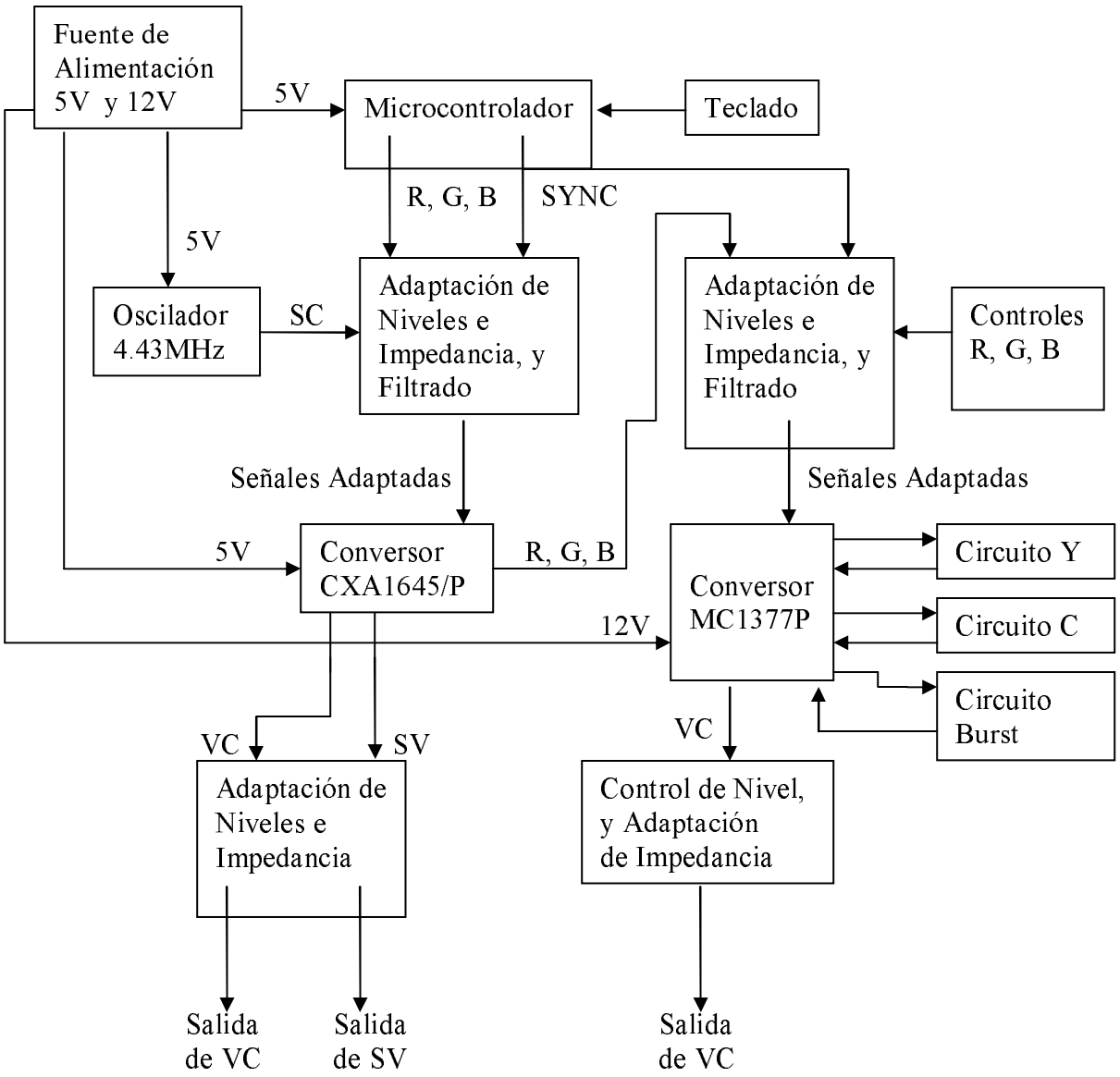


Figura 3.1 – Diagrama de bloques de Generador de Vídeo.

- **Teclado:** Consiste en un simple pulsador con una resistencia de pull-up. Su misión es cambiar los diferentes modos de funcionamiento.

- **Oscilador de 4.43MHz:** Genera la subportadora de color de 4.43MHz necesaria para el conversor de vídeo CXA1645/P. Está construido con inversores TTL, exactamente con el circuito integrado 74HC04N.

- **Adaptación de Niveles e Impedancia, y Filtrado (1):** Son redes que adaptan los niveles lógicos TTL del Microcontrolador a niveles de 1Vpp necesarios en las entradas de componentes de color del conversor de vídeo CXA1645/P, cuidando además que las impedancias de entrada sean las adecuadas. También se incluyen redes de filtrado para la señal de sincronismo y la subportadora de color, que eliminan en gran medida las componentes de alta frecuencia (ruido).

- **Conversor CXA1645/P:** Es un circuito integrado de SONY, que utiliza la señal de sincronismo compuesto junto con las señales de componentes de color para generar la señal de vídeo compuesto. También dispone de salidas de luminancia y crominancia (súper vídeo). No incluye oscilador para generar la subportadora de color, por lo que también será necesario añadirla tal como se ha mencionado antes.

- **Adaptación de Niveles e Impedancia:** Estas redes desacoplan la componente continua de la señal de vídeo, y aseguran una impedancia de salida de 75Ω.

- **Adaptación de Niveles e Impedancia, y Filtrado (2):** Son redes que adaptan los niveles de las salidas en cascada del conversor CXA1645/P, a niveles de 1Vpp necesarios en las entradas de componentes de color del conversor de vídeo MC1377P, cuidando además que las impedancias de entrada sean las adecuadas. También se incluye una red de filtrado para la señal de sincronismo que elimina posibles ruidos.

- **Controles R, G, B:** Consisten en una serie de interruptores que permiten eliminar de forma independiente cada una de las componentes de color.

- **Conversor MC1377P:** Es un circuito integrado de Motorola, que utiliza la señal de sincronismo compuesto junto con las señales de componentes de color para generar la señal de vídeo compuesto. Incluye oscilador para generar la subportadora de color, pero en cambio se ha de añadir cierta circuiteria externa para otras funciones como: filtro de color, retardo de luminancia, control de burst. La exteriorización de estos circuitos nos va a permitir un posible control de los mismos tal como se verá más adelante.

- **Circuito Y:** Incluye una red de retardo para la señal de luminancia, necesaria para la coincidencia de esta con la señal de crominancia (se verá con más detalle en el siguiente capitulo). También incluye un mando que permite eliminar la señal de luminancia.

- **Circuito C:** Consiste en un filtro para la señal de crominancia (se verá con más detalle en el siguiente capitulo). La señal de crominancia puede ser eliminada mediante un interruptor existente para tal propósito.

- **Circuito Burst:** Este circuito ajusta la posición de la salva de color (se verá con más detalle en le siguiente capitulo). Se han incluido mandos para el control de la misma.

- **Control de Nivel y Adaptación de Impedancia:** Ajusta el nivel de salida de la señal de vídeo compuesto, y al mismo tiempo asegura que la impedancia de salida sea la estándar (75Ω).

3.2. DESCRIPCIÓN DEL MONITOR.

En el diagrama de bloques de la figura 3.2 se muestran las diferentes partes que constituyen el Monitor.

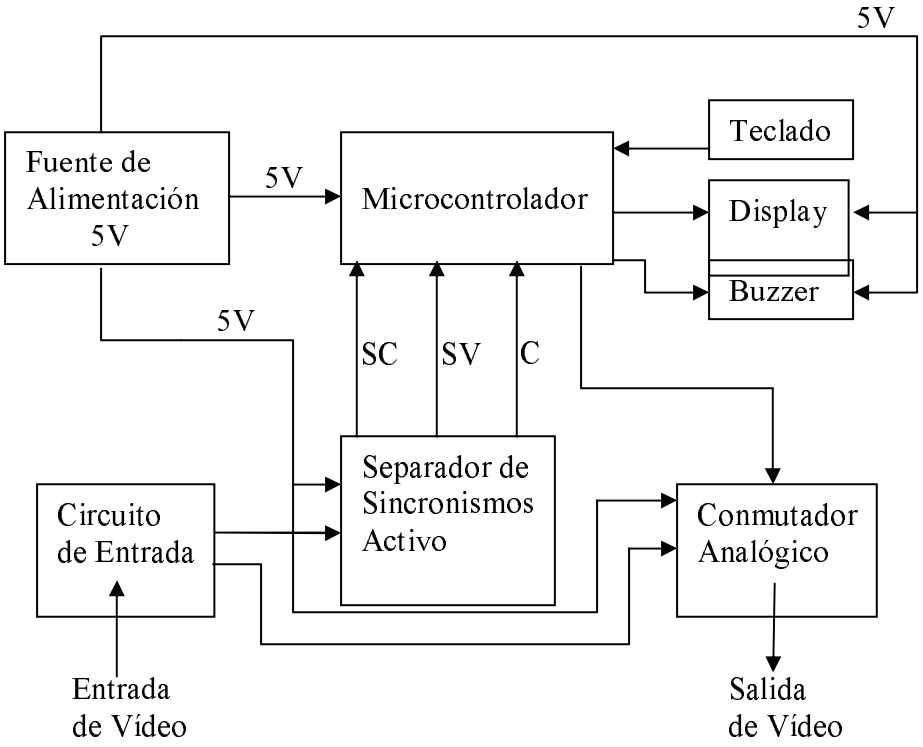


Figura 3.2 – Diagrama de bloques de Monitor de Vídeo.

En el diagrama de bloques del Monitor se pueden observar los siguientes módulos:

- **Fuente de Alimentación:** La fuente de alimentación es común para el Generador y el Monitor, y su estructura es la descrita en el apartado anterior.

- **Microcontrolador:** El microcontrolador es un PIC16F876-4 de Microchip. Recibe información del teclado (modo de funcionamiento, campo, línea) y del separador de sincronismos activo (sincronismo compuesto, sincronismo vertical, campo), y manda la información que corresponda al display (muestra menús e información), buzzer (proporciona alerta sonora) y conmutador analógico (deja pasar la señal de vídeo).

- **Teclado:** El teclado esta formado por una matriz de 5 pulsadores, y utiliza las resistencias de pull-up que incorpora el microcontrolador. Su función es la de seleccionar las opciones que correspondan en los menús mostrados en el display.

- **Display:** El display es un LM041L de Hitachi. Consta de 4 líneas de 16 caracteres. En el se muestran las opciones disponibles, así como la información correspondiente al modo de funcionamiento seleccionado.

- **Buzzer:** Proporciona una alerta sonora para informar de los sucesos importantes que se producen en el sistema. Está construido con un pequeño altavoz y un amplificador que lo pilota, encargándose el microcontrolador de generar la señal alterna que ataca al amplificador.

- **Circuito de Entrada:** Consiste en una carga estándar de 75Ω , junto con el desacoplo de continua de la señal. De aquí se obtiene la señal para el conmutador analógico, y para el separador de sincronismos activo, estando este ultimo precedido por un filtro que favorece la separación de la señal de vídeo compuesto.

- **Separador de Sincronismos Activo:** Se trata del circuito integrado LM1881N de National Semiconductor. Este circuito recibe la señal de vídeo compuesto y extrae de ella la señal de sincronismo compuesto, suministrando además señales digitales que informan del sincronismo vertical y del campo. Estas señales de salida (utilizadas por el microcontrolador) son la base del sistema (se verá con más detalle en el siguiente capitulo).

- **Conmutador Analógico:** Su función es la de permitir el paso de la señal de vídeo en el instante adecuado (según le indique el microcontrolador), para de esta forma visualizar en el osciloscopio solo la línea deseada. Se trata del circuito integrado HEF4066BP del cual solo se utiliza uno de los cuatro conmutadores disponibles.

3.3. DESCRIPCIÓN Y CONSTRUCCIÓN DE GENERADORES DE VÍDEO - AFICIONADOS.

Con estas unidades se realizaron multitud de pruebas que sirvieron de practicas, complementando así a los conocimientos teóricos adquiridos con libros de teoría, libros de practicas, y programas de simulación.

3.3.1. GENERADOR DE VÍDEO – AFICIONADO Nº1.

Se trata de un equipo que muestra una carta de barras a color centrada en la pantalla, un texto deslizante en la parte superior, y un texto parpadeante en la parte inferior. En el texto deslizante se puede leer "SORRY FOR BUGS IN VXX : GOOD LUCK WITH THIS V05 BY", mientras que el texto parpadeante dice "IW2KGH", siendo este ultimo el indicativo del aficionado (una imagen del mismo se puede ver en la figura 3.3).

Figura 3.3 – Generador de vídeo-aficionado nº1.

El generador está constituido principalmente por dos circuitos integrados:

1) Un microcontrolador PIC16F84, encargado de la generación de las señales de vídeo (sincronismo compuesto, R, G, y B), una señal de audio, y el control del teclado.
2) Un conversor de vídeo MC1377, que genera la señal de vídeo-compuesto.

El teclado consta de dos pulsadores y un interruptor, con los cuales se puede cambiar el color de fondo de los mensajes, y el texto del mensaje deslizante.

También se dispone de mandos adicionales para el control del nivel de la salida de video (vídeo out) y de la salida de audio (tone out), así como el control de la posición de la salva de color.

La alimentación del circuito es de 12VDC. El conversor de vídeo se alimenta directamente a través de un diodo de protección junto con unos condensadores de filtrado, mientras que el microcontrolador se alimenta con 5VDC a través de un regulador de tensión integrado del tipo 7805.

Todo lo expuesto anteriormente se puede apreciar en el esquema del circuito representado en la figura 3.4.

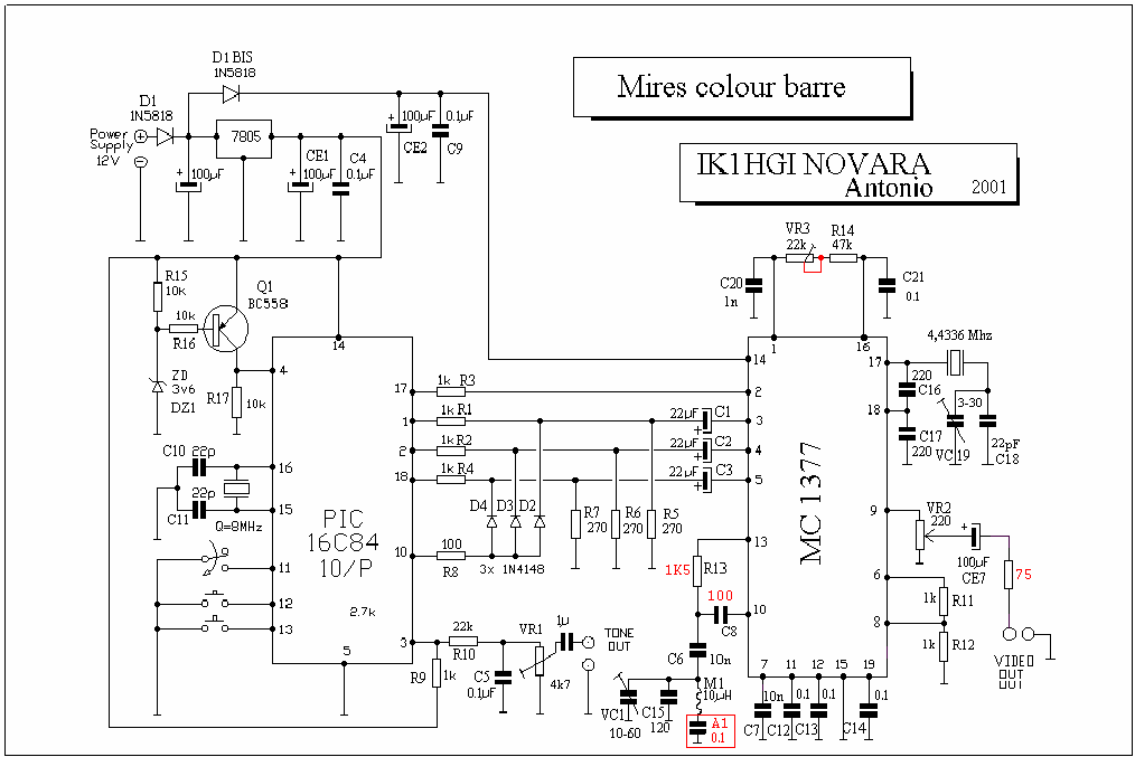


Figura 3.4 - Esquema del generador de vídeo-aficionado nº1.

Para la construcción del prototipo se utilizó la placa de circuito impreso del autor del proyecto. La placa está realizada a una cara, y todos los procesos de elaboración de la misma se comentaran más adelante en el capítulo 6. La placa de circuito impreso, la situación de los componentes, y la lista de componentes se muestran en las figuras 3.5, 3.6, y 3.7 respectivamente.

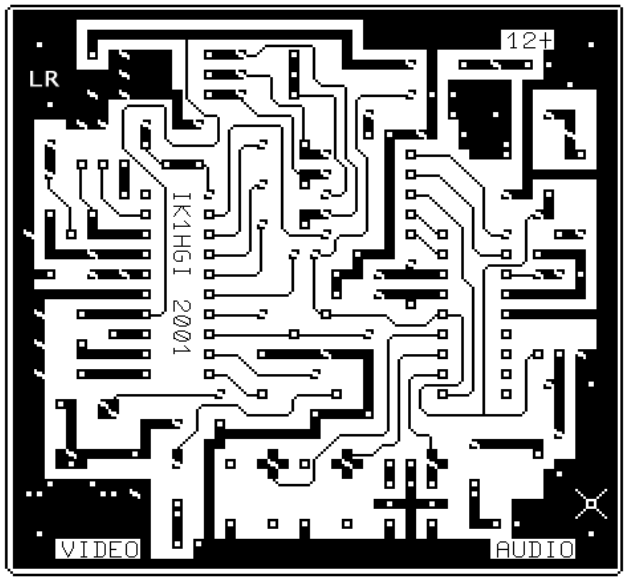


Figura 3.5 - PCB del generador de vídeo-aficionado nº1.

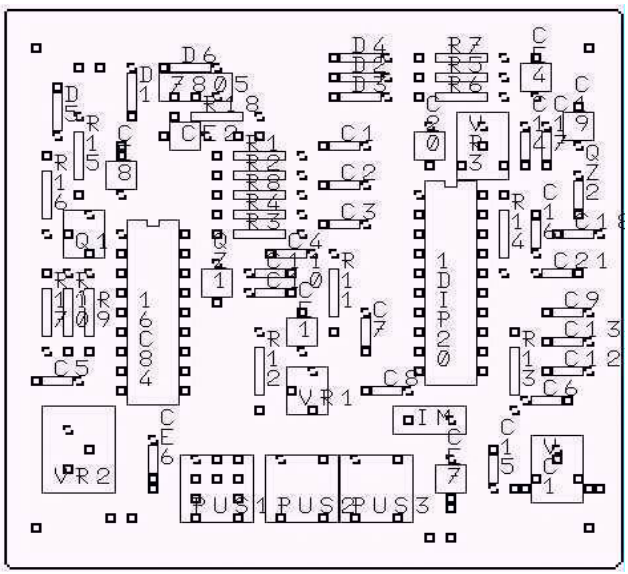


Figura 3.6 - Componentes del generador de vídeo-aficionado nº1.

1DIP18 PIC16CóF84-10/P	IM 10uH
1DIP20 MC1377	PUS1 P1
1TO220 7805	PUS2 P2
C1 22uF/16V	PUS3 P3
C2 22uF/16V	Q1 BC308
C3 22uF/16V	XT1 8MHz
C4 0.1uF	XT2 4.433MHz
C5 33nF	R1 1K
C6 10nF	R2 1K
C7 10nF	R3 1K
C8 0.1uF	R4 1K
C9 0.1uF	R5 10K
C10 27pF	R6 3.3K
C11 27pF	R7 100
C12 0.1uF	R8 100
C13 0.1uF	R9 1K
C14 220pF	R10 22K
C15 120pF	R11 1K
C16 220pF	R12 1K
C17 220pF	R13 1K
C18 22pF	R14 10K
C19 10-30pF	R15 47K
C20 1nF	R16 10K
C21 0.1uF	R17 10K
CE1 10uF/16V	R18 470
CE2 47uF/25V	D1 1N5818
CE4 1uF	D2 1N4148
CE6 1uF	D3 1N4148
CE7 470uF/16V	D4 1N4148
CE8 1uF	D5 DZ3.9V
VC1 7-37	D6 1N4007
VR1 220	
VR2 220	
VR3 22K	

Figura 3.7 – Lista de componentes del generador de vídeo-aficionado nº1.

El software de la aplicación está escrito en lenguaje ensamblador con un editor de texto estándar, y se ensambló con la herramienta de Microchip "MPASM for Windows". Para cargar la memoria eeprom del microcontrolador se utilizó el programa "ICProg", junto con el programador "TE-20".

A continuación se muestra el código fuente de la aplicación:

```
      TITLE   "MONOPIC_COLO"        ;PICDREAM modificato da IW2KGH
                                    ;Salvatore

      list p=16C84,  f=inhx32       ;Enter device name
      __config 0x3FF2               ;configurazione (fuse_off )

#Define     W       0
#Define     F       1
#define     Sync    PortA,0   ; Sincronismo video uscita RA0  (PIN 17)
#define     Audio   PortA,4   ; Sincronismo video uscita RA4  (PIN 3)

LUNCAR      EQU   D'9'
ULTICAR     EQU   D'62'-LUNCAR;Ultimo carattere del circolare
PRIMCIR     EQU   D'0'        ; Primo carattere della scritta circolare
ULTICIR     EQU   D'52' ; Ultimo carattere della scritta circolare
PRIMFIS     EQU   D'53'       ; Primo carattere della scritta fissa
ULTIFIS     EQU   D'61'       ; Ultimo carattere della scritta fissa
CELLSP      EQU   D'0'        ;cella da sporcare in caso di :::::

      cblock  0x00
            INDF,RTCC,PCL,Status,FSR,PortA,PortB
      endc
      cblock  0x08
            EEData,EEAdr,PClath,IntCon
      endc
      cblock  0x08
            EECON1,EECON2,PClath,IntCon
      endc
      cblock  0x00
            C,DC,Z,PD,TO,RP0,RP1,IRP
      endc
      cblock  0x00
            RD,WR,WREN,WRERR,EEI
      endc

; Page 1 registers
TRISA       EQU   05H
PORTA EQU   05H
Roption     EQU   01H
Timer0            EQU   01H
Intcon            EQU   0BH
TrisA       EQU     05H
TrisB             EQU     06H
RAMbase     EQU     0CH

;DNOP - doppia NOP. Delay di 2 cycles

DNOP    MACRO
      LOCAL   Label
Label   GOTO    Label+1
      ENDM

SKIPNC  MACRO
      BTFSC   Status,C
      ENDM

SKIPSC  MACRO
      BTFSS   Status,C
      ENDM
```

```
SKIPNZ  MACRO
      BTFSC   Status,Z
      ENDM

SKIPSZ  MACRO
      BTFSS   Status,Z
      ENDM

      LIST

      CBLOCK  RAMbase                                 ;36 celle
            TIME,CNT_0                                ;2    2
            Count2,Count3,SubSec                      ;3    5
            HrT,HrU,MiT,MiU,SeU                       ;5    10
            CA0,CA1,CA2,CA3,CA4,CA5,CA6,CA7,CA8 ;9    19
            Ta0,Ta1,Ta2,Ta3,Ta4,Ta5,Ta6,Ta7,Ta8        ;9    28
            Ptrtxt,CNT_1,CNT_R                        ;3    31
            CNTCA,TABCA,STATO,DISPE,PUNCA             ;5    36

;celle ram aggiuntive per micro 16F84
;ora non necessarie

      ENDC

;TIME       cella utilizzata dal delay (nel ciclo)
;CNT_0  utilizzato chiamate per sapere quante righe video fare e altro
;Count2     Contatore di quadri per azione con pulsanti (riservata )
;Count3     Contatore per velocit… di scorrimento
;SubSec     utilizzata per semiquadri a/b e orologio 1sec
;Ptrtxt  punt. al primo carattere per display scritta scorrevole e no
;CNT_1   numero di linee che definiscono il punto carattere (DISPLAY)
;CNT_R      numero di riga del carattere in display (DISPLAY)
;CNTCA      puntatore carattere circolare durante memorizzazione
;(stato0 e 1)
;TABCA      carattere da memorizzate convertito ascii_tab
;STATO      stato per sapere quale comando devo eseguire
;DISPE      tipo di display a salti o punti e velocit… da sommare
;PUNCA      punt. carattere scritta fissa durante memorizzazione frase

;*****************************************************************
;  EEPROM
;*****************************************************************
      ORG     2100
;scritta scorrevole (il primo CarSP va lasciato )

;frase: " SORRY FOR BUGS IN VXX : GOOD LUCK WITH THIS V05  BY"

      DT      CarJ,CarO,CarS,CarE,CarSP,CarJ,CarU,CarA          ;8
      DT      CarN,CarSP,CarB,CarA,CarU,CarT,CarI,CarS          ;8 16
      DT      CarT,CarA,CarSP,CarSP,CarI,CarPS,CarT,CarPS       ;8 24
      DT      CarI,CarN,CarD,CarU,CarS,CarT,CarR,CarI           ;8 32
      DT      CarA,CarL,CarSP,CarSP,CarE,CarPS,CarP,CarPS       ;8 40
      DT      CarS,CarPS,CarSP,CarL,CarA,CarSP,CarR,CarA        ;8 48
      DT      CarB,CarI,CarD,CarA,CarFI                         ;5 53
```

```
;scritta fissa

      DT      CarU,CarPS,CarH,CarU,CarE,CarL,CarV,CarA,CarSP     ;9 62

;Dati di gestione per il programma non modificare

      DE      42,0EC                                             ;2 64

;*****************************************************************
;COLORE:cella 62 EEPROM 'XXXX XXXX'
;                            ||||___          Colore di sfondo sopra
;                        ||||_____            Colore di fondo sotto
;
;MODI:      cella 63 EEPROM 'XXXX XXXX'
;                        |||| ||||___          Velocit… di scorrimento
;                        ||||_____
;                        |||_____             Tono:  1=singolo   0=bitonale
;                        ||_____              Fondo: 1=alternanza 0=fissa
;                        |_____               Fondo: 1=orologio   0=fissa

;*****************************************************************
;       START  del programma
;*****************************************************************
      ORG     0
      GOTO    INIZIO
      ORG     4
      RETURN

;*****************************************************************
;       TABELLA definizione matrice dei caratteri
;*****************************************************************
;Table of caracters
Table ADDWF CNT_R,W
      ADDWF PCL,F
Tbase   equ     $
Car0    equ     $-Tbase
CarO    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
Car1    equ     $-Tbase
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00001100'     ; ....**..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00001110'     ; ....***.
Car2    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000010'     ; ......*.
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00001000'     ; ....*...
      RETLW   B'00011111'     ; ...*****
```

```
Car3    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ;....*...*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000110'     ; .....**.
      RETLW   B'00000001'     ; .......*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
Car4    equ     $-Tbase
      RETLW   B'00000010'     ; ......*.
      RETLW   B'00000110'     ; .....**.
      RETLW   B'00001010'     ; ....*.*.
      RETLW   B'00010010'     ; ...*..*.
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00000010'     ; ......*.
      RETLW   B'00000010'     ; ......*.
Car5    equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00011110'     ; ...****.
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
Car6    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
Car7    equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000010'     ; ......*.
      RETLW   B'00000010'     ; ......*.
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
Car8    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
Car9    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001111'     ; ....****
      RETLW   B'00000001'     ; .......*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ; ....***.
```

```
CarSP   equ     $-Tbase
      RETLW   B'00000000'     ; ........
CarFI   equ     $-Tbase
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
CarDP   equ     $-Tbase
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000000'     ; ........
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
CarPS   equ     $-Tbase
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00000000'     ; ........
      RETLW   B'00010000'     ; ...*....
CarA    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00011111'     ; ...*****
CarH    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00011111'     ; ...*****
CarU    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001110'     ;.....***.
CarD    equ     $-Tbase
      RETLW   B'00011110'     ; ...****.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
CarB    equ     $-Tbase
      RETLW   B'00011110'     ; ...****.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
CarP    equ     $-Tbase
      RETLW   B'00011110'     ; ...****.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00011110'     ; ...****.
```

```
CarL    equ     $-Tbase
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
CarE    equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00011100'     ; ...***..
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
CarF    equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00011100'     ; ...***..
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
CarJ    equ     $-Tbase
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00000001'     ; .......*
      RETLW   B'00010001'     ; ...*...*
CarG    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010011'     ; ...*..**
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
CarQ    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010101'     ; ...*.*.*
      RETLW   B'00010011'     ; ...*..**
CarS    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00000001'     ; .......*
      RETLW   B'00010001'     ; ...*...*
CarC    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010000'     ; ...*....
      RETLW   B'00010001'     ; ...*...*
```

```
CarI    equ     $-Tbase
      RETLW   B'00001110'     ; ....***.
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00001110'     ; ....***.
CarK    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010010'     ; ...*..*.
      RETLW   B'00010100'     ; ...*.*..
      RETLW   B'00011000'     ; ...**...
      RETLW   B'00010100'     ; ...*.*..
      RETLW   B'00010010'     ; ...*..*.
CarM    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00011011'     ; ...**.**
      RETLW   B'00010101'     ; ...*.*.*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
CarN    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00011001'     ; ...**..*
      RETLW   B'00010101'     ; ...*.*.*
      RETLW   B'00010011'     ; ...*..**
CarY    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001010'     ; ....*.*.
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
CarR    equ     $-Tbase
      RETLW   B'00011110'     ; ...****.
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00011110'     ; ...****.
      RETLW   B'00010100'     ; ...*.*..
      RETLW   B'00010010'     ; ...*..*.
      RETLW   B'00010001'     ; ...*...*
CarV    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
CarX    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00001010'     ; ....*.*.
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00001010'     ; ....*.*.
```

```
CarW    equ     $-Tbase
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010001'     ; ...*...*
      RETLW   B'00010101'     ; ...*.*.*
      RETLW   B'00011011'     ; ...**.**
      RETLW   B'00010001'     ; ...*...*
CarZ    equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00000001'     ; ......  *
      RETLW   B'00000010'     ; .....  *.
      RETLW   B'00000100'     ; ....  *..
      RETLW   B'00001000'     ; ....*...
      RETLW   B'00010000'     ; ...*....
CarT    equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
      RETLW   B'00000100'     ; .....*..
CarBL   equ     $-Tbase
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00011111'     ; ...*****
      RETLW   B'00011111'     ; ...*****

;*****************************************************************
;    inizio del programma e settaggio del PIC e EEPROM
;*****************************************************************
WEEINI      CALL  MEMEEP      ;*
      MOVLW D'254'            ;*254    Ritardo per poter scrivere
      MOVWF CNT_0             ;*          la EEPROM a bordo del PIC
      CALL  BLKLINE           ;*          nessuna operazione su EEPROM
      RETURN                  ;*          per almeno 14 millisecondi

INIZIO
      BSF     Status,RP0      ;     select bank 1
      MOVLW B'00000000'       ;     Protezione per scrittura in
      MOVWF EECON1            ;     EEPROM all'accensione
      BCF     EECON1,WREN     ;      ( spero & spero )
      MOVWF EECON2            ;
      MOVLW   B'01110'        ;     Port_A  0=outputs  1=inputs
      MOVWF   TrisA           ;
;     MOVLW B'00000000'       ;     serve per testare con MPLAB
      MOVLW   B'11100000'     ;      Port_B  0=outputs  1=inputs
      MOVWF   TrisB           ;
      MOVLW   B'00000011'     ;pull up on, interup \_, internal clok,
                              ;inc rb4  _/
      MOVWF       Roption     ;     prescaler TMR0 / 2
      BCF     Status,RP0      ;     select bank 0
```

```
;????????????????????????????????????????????????????????????????????
;
;INSEPROVE
;
;????????????????????????????????????????????????????????????????????

      MOVLW   D'255'          ;*
      MOVWF CNT_0             ;*    solo ritardo iniziale
      CALL    BLKLINE         ;*

      MOVF    PortB,W         ;*
      ANDLW   B'11000000'     ;*          pulsanti entrambi pigiati
      SKIPSZ                  ;*
      GOTO    INIZ1           ;*    se si inizzializzo la EEPROM

      BTFSS   PortB,5         ;     in funzione dell'interruttore WR
      GOTO  RESET2

      MOVLW CarX              ;*
      MOVWF   EEData          ;*
      CLRF    EEAdr           ;*    scrivo delle X in tutta la EEPROM
INIEE CALL  WEEINI            ;*
      INCF  EEAdr,1           ;*
      BTFSS EEAdr,6           ;*
      GOTO  INIEE             ;*

      MOVLW D'4'              ;
      MOVWF EEAdr             ;
      MOVLW CarSP             ;
      MOVWF EEData            ;
      CALL  WEEINI            ;
      INCF  EEAdr,1           ;*
      MOVLW CarV              ;
      MOVWF EEData            ;
      CALL  WEEINI            ;
      INCF  EEAdr,1           ;*
      MOVLW Car0              ;
      MOVWF EEData            ;
      CALL  WEEINI            ;
      INCF  EEAdr,1           ;*
      MOVLW Car5              ;     Versione 5
      MOVWF EEData            ;
      CALL  WEEINI            ;
      INCF  EEAdr,1           ;*
      MOVLW CarFI             ;*    metto punto di ripetizione
      MOVWF EEData            ;*    circolare alla scritta
      CALL  WEEINI            ;*

RESET2
      MOVLW D'62'             ;
      MOVWF EEAdr             ;*    sistemo cella di servizio
      MOVLW 042               ;*
      MOVWF EEData            ;*
      CALL  WEEINI            ;*

      INCF  EEAdr,1           ;*
      MOVLW 0EC               ;*    sistemo cella di servizio
      MOVWF EEData            ;*
      CALL  WEEINI            ;*
```

```
INCF  EEAdr,1                ;*
      MOVLW CarDP            ;*    sistemo 1 cella della scorrevole
      MOVWF EEData           ;*
      CALL  WEEINI           ;*

INIZ1 CLRF    SeU            ;     azzero clock a 00.00
      CLRF    MiU            ;
      CLRF    MiT            ;
      CLRF    HrU            ;
      CLRF    HrT            ;
      CLRF  Ptrtxt           ;     puntatore frase scorrevole
      CLRF    Count3         ;
      CLRF    DISPE          ;
      CLRF  STATO            ;
      BSF   STATO,4          ;
      MOVLW CELLSP           ;
      MOVWF EEAdr            ;

;*********************************************************************
;   Programma principale o loop main
;*********************************************************************
FRAME                        ;
      CALL  EQUALIZ          ;     totale 7,5 linee

;*****  BLACK LINES   ************ line 0 (+35)=35

      MOVLW   D'35'          ; 1
      MOVWF CNT_0            ; 2
      CALL    BLKLINE        ;128

;**** INIT TEXTE     ***************     line 35(+3)=38

      Call    PREREAD        ;3

;****  PULSANTI    ****************      line 38(+1)=39

      Call    PULSANTI       ;1

;**** DISPLAY TEXTE ************   line 39 (+35)=75

      MOVLW B'00000000'      ;1  0xxxxxx1= scorrevole e sopra
      MOVWF DISPE            ;2
      CALL    DISPLAY        ;4    128

;**** BLACK LINES ***************   line 75 (+15)= 90

      MOVLW   D'15'          ; 1
      MOVWF CNT_0            ; 2
      CALL    BLKLINE        ; 128

;***** GREY BARS ***************  line 90 (+120)=210

      MOVLW   D'120'         ;1
      MOVWF   CNT_0          ;2
      CALL BARRE             ;128
```

```
;**** BLACK LINES   **************  line 210 (+14)=224

      MOVLW   D'14'           ; 1
      MOVWF CNT_0             ; 2
      CALL    BLKLINE         ;128

;***** PREPARE CLOCK ***********    line 224 (+3)=227

      CALL    PREPH           ;3

;**** DISPLAY CLOCK 57 line  **** * line 227 (+36)=263

      MOVLW B'11000001'       ;1   1xxxxxx1=fissa  fondo
      MOVWF DISPE             ;
      CALL    DISPLAY         ;1    128

;***** INCREMENT TIME **********    line 263 (+1)=264

INCTIM      CALL    INCTIME   ;2    128

; ***** BLACK LINES *************  line 264 (+41)=305

      MOVLW   D'41'           ; 1
      MOVWF CNT_0             ; 2
      CALL    BLKLINE         ;128

      GOTO  FRAME             ;RITORNO A CAPO

;************************************************************
;     Equalizzazione 7,5 + 305 visible Lines =312,5 *2 =625
;   inserisce linee per l'interlacciamento
;   e ritorno a capo del video secondo norme CCIR/PAL
;************************************************************
EQUALIZ
      CALL  SINCK             ;14     18
      MOVLW   4               ;1
      BTFSS   SubSec,0        ;1-2
      MOVLW   5               ;1
      MOVWF   CNT_0           ;1     22
Loop1   CALL      RIT_46      ;46   68  (64  4)
      CALL  SINCK             ;14   18
      NOP                     ;1
      DECFSZ  CNT_0,1         ;1-2
      GOTO    Loop1           ;2     22

      CALL  RIT_48            ;48   69   (64 5 )
      NOP                     ;1     6

; 5 mezze righe per Equalizzazione

      BCF     Sync            ;1   7   ;30us Sync
      CALL  AUDIO             ;8

      MOVLW   4               ;16
      MOVWF   CNT_0           ;17
```

```
Loop2
      CALL  RIT_48            ;48  65
      BSF     Sync            ;1    66  ;2us Nero
      CALL  AUDIO             ;8
      BCF     Sync            ;1    7     ;30us Sync
      DNOP                    ;2    9
      NOP                     ;1    10
      DECFSZ CNT_0,1          ;  -
      GOTO    Loop2           ;13

      CALL  RIT_48            ;52   64
      NOP                     ;1
      BSF     Sync            ;2          ;2us Nero

; 5 o 4 mezze righe di equalizzazione per interlacciare

      CALL  DSINCK            ;16   20
      MOVLW   4               ;1
      BTFSC   SubSec,0        ;1-2
      MOVLW   3               ;1
      MOVWF   CNT_0           ;1     22
Loop3   CALL      RIT_46      ;46   68  (64  4 )
      CALL  SINCK             ;14   18
      NOP                     ;1
      DECFSZ  CNT_0,1         ;1-2
      GOTO    Loop3           ;2    22

      MOVLW D'12'             ;     ritardo
      CALL  RIT_P0            ;41  62

      RETURN                  ;2    64

;************************************************************
;    routines di ritardo ( delay )
; es: RIT_51 = ritardo di 51 Time inclusi chiamata e ritorno
;************************************************************
RIT_48      DNOP              ;
RIT_46      MOVLW D'13'       ;
      GOTO  RIT_P0            ;

RIT_P2      NOP               ;
RIT_P1      NOP               ;
RIT_P0      MOVWF TIME        ;     ritardo  2  + 3*W + 2
RITAY DECFSZ      TIME,1      ;           call      return
      GOTO  RITAY             ;
      RETURN                  ;

RIT_8 NOP                     ;
RIT_7 NOP                     ;
RIT_6 NOP                     ;
RIT_5   NOP                   ;
RIT_4   RETURN                ;

DSINCK      DNOP              ;2
SINCK BCF   Sync              ;1    Genera inpulso di sincronismo
      CALL  AUDIO             ;6
      BSF   Sync              ;1    toglie sinc
      RETURN                  ;2  (12) -(14)
```

```
AUDIO
      BTFSC Timer0,7          ;
      BSF   PORTA,4           ;
      BTFSS Timer0,7          ;     per 9 time= 4,5 micro_secondi
      BCF   PORTA,4           ;
      RETURN                  ;

;*************************************************************
;     Righe nere sul televisore
;  il numero di righe , definito in  CNT_0 prima della chiamata
;*************************************************************
BLKLI_0
      CALL  RIT_5             ;5   1 4

BLKLINE     CALL  SINCK       ;14  18   impulso di sincronismo
      MOVLW D'33'             ;                 ritardo
      CALL  RIT_P2            ;106  124

      DECFSZ      CNT_0,1     ;1-2  125  126
      GOTO  BLKLI_0           ;2     127   -
      RETURN                  ;2     128

;*************************************************************
;       BARRE:  dei grigi in verticale sul video
;  il numero di righe utilizzato per le barre e definito in CNT_0
;*************************************************************
BARRE_0     CALL  RIT_5       ;5  1-4
BARRE CALL  SINCK             ;14  18

      MOVLW D'2'              ;           ritardo
      CALL  RIT_P1            ;9  27

      MOVLW D'7'              ;1          numero di barre
      MOVWF CNT_1             ;1          in CNT_1
      MOVLW D'15'             ;1          pongo in uscita il valore 15
      IORWF PORTA,1           ;1
      MOVLW B'00000'          ;1          Port_A definisco gli
      TRIS  PORTA             ;1          output per i grigi
      DNOP                    ;2   35
BARRE_1     CALL  RIT_6       ;6          genero le barre partendo dal
      MOVLW -D'2'             ;1          bianco (parto 15 e sommo -2 )
      ADDWF PORTA,1           ;1
      DECFSZ      CNT_1,1     ;1-2
      GOTO  BARRE_1           ;2
      CALL  RIT_7             ;
      MOVLW B'01110'          ;1   123
      TRIS  PORTA             ;1   124
      NOP                     ;
      DECFSZ      CNT_0,1     ;1-2 125
      GOTO  BARRE_0           ;2   127
      RETURN                  ;2   128
```

```
;*****************************************************************
;     DISPLAY: mette sul video i caratteri prendendoli
;                dalla EEPROM
;*****************************************************************
DISPLAY     CALL  SINCK       ;14  18
      MOVLW D'7'              ;           Numero di punti per carattere
      MOVWF CNT_0             ;           metto in CNT_0
      MOVLW H'0FF'            ;           numero di linea del carattere
      MOVWF CNT_R             ;22   in display
      MOVLW   D'63'           ;1
      CALL    EEREADW         ;9   32     leggo parola di controllo da
      BTFSS EEData,3          ;1          se devo scorrere a salti o
      BSF   DISPE,7           ;1   34     a pixel
      MOVLW   D'62'           ;1   35
      CALL    EEREADW         ;9   44 leggo la parola di controllo da

      BTFSS DISPE,0           ;1      testo se scritta sopra o sotto
      SWAPF EEData,1          ;1          per eseguire swap
      RLF   EEData,W          ;1          i bit di colore e li pongo
      ANDLW B'00001111'       ;1          in dispe
      IORWF DISPE,1           ;1   49
      RLF   Count3,W          ;1
      ANDLW B'11110000'       ;1
      BTFSS DISPE,7           ;1
      IORWF DISPE,1           ;1   53
      MOVLW CELLSP            ; zzz
      MOVWF EEAdr             ;
      MOVLW D'22'             ;           ritardo
      CALL  RIT_P1            ;72 127

INCLIN
      MOVLW D'4'              ;      128
      MOVWF CNT_1             ;1          linee per punto
      INCF    CNT_R,1         ;1    2     inc riga per prossimo giro
      CALL  DSINCK            ;16  18

      MOVF    CA0,W           ;1    Prendo Il carattere lo sommo alla
      CALL    Table           ;7    riga e vado a prendere il segmento
      MOVWF   Ta0             ;1    27    equivalente lo metto in ram
      MOVF    CA1,W           ;1
      CALL    Table           ;7
      MOVWF   Ta1             ;1   36
      MOVF    CA2,W           ;1
      CALL    Table           ;7
      MOVWF   Ta2             ;1   45
      MOVF    CA3,W           ;1
      CALL    Table           ;7
      MOVWF   Ta3             ;1  54
      MOVF    CA4,W           ;1
      CALL    Table           ;7
      MOVWF   Ta4             ;1  63
      MOVF    CA5,W           ;1
      CALL    Table           ;7
      MOVWF   Ta5             ;1  72
      MOVF    CA6,W           ;1
      CALL  Table             ;7
```

```
      MOVWF   Ta6             ;1  81
      MOVF    CA7,W           ;1
      CALL  Table             ;7
      MOVWF   Ta7             ;1  90
      MOVF    CA8,W           ;1
      CALL  Table             ;7
      MOVWF   Ta8             ;1  99
      MOVLW D'7'              ;           ritardo
      CALL  RIT_P0            ;26 125

SLINE
      CALL  RIT_7             ;7  128 4
      CALL  SINCK             ;14  18
      CALL  RIT_7             ;7  25
      CALL  RIT_5             ;5  30

      MOVF  DISPE,W           ;1
      ANDLW B'00001110'       ;1
      IORWF PORTA,1           ;1
      MOVLW B'00000'          ;1          in funzione bit x di EEData
      BTFSS EEData,3          ;1          i bit di port_A output
      TRIS  PORTA             ;1  36    o li lascio come input
      BTFSS DISPE,6           ;  |
      CALL  RIT_5             ;   |
      BTFSC DISPE,5           ;    |      in testa e in coda
      GOTO  RITCAR1           ;     |     per lo scorrimento a pixel
      DNOP                    ;      )    due volte vale 17
      NOP                     ;     |
RITCAR1     BTFSS DISPE,4     ;    |
      DNOP                    ;   |  ES ( 10+7 )
                              ;46
      MOVF    Ta0,W           ;1    Carac 1
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1 7
      MOVF    Ta1,W           ;1    Carac 2
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
      MOVF    Ta2,W           ;1    Carac 3
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
      MOVF    Ta3,W           ;1    Carac 4
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
```

```
      MOVF    Ta4,W           ;1    Carac 5
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
      MOVF    Ta5,W           ;1    Carac 6
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
      MOVF    Ta6,W           ;1    Carac 7
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
      MOVF    Ta7,W           ;1    Carac 8
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1  7
      MOVF    Ta8,W           ;1    Carac 9
      MOVWF   PortB           ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      RLF     PortB,1         ;1
      CLRF    PortB           ;1 109
      BTFSC DISPE,6           ;     |
      CALL  RIT_5             ;      |
      BTFSS DISPE,5           ;       |          in testa e in coda
      GOTO  RITCAR2           ;        |  per lo scorrimento a pixel
      DNOP                    ;         )       due volte vale 21
      NOP                     ;        |
RITCAR2     BTFSC DISPE,4     ;       |
      DNOP                    ;11  |      ES ( 10+11 )
                              ;120
      MOVLW B'01110'          ;1  121pongo i bit di port_A come input
      TRIS  PORTA             ;1  122     quelli barra dei grigi
      DECFSZ  CNT_1,1         ;1-2  123  124
      GOTO    SLINE           ;2     125

      DECFSZ  CNT_0,1         ;1-2  125  126
      GOTO    INCLIN          ;2     127
      RETURN                  ;2     128
```

```
;*********************************************************************
;       PULSANTI: lettura dei comandi eseguiti tramite pulsanti
;         STATO_0  set orologio e set scritta / orologio
;         STATO_1  modifica della scritta scorrevole
;         STATO_2  modifica della scritta fissa
;         STATO_3  cambio velocit… di scorrimento scritta
;         STATO_4
;*********************************************************************
PULSANTI
      CALL  DSINCK            ;18

      BTFSC   PortB,5         ;1      in funzione dell'interruttore WR
      CLRF  STATO             ;1          Impongo lo stato da eseguire
      BTFSS   PortB,5         ;1
      BSF   STATO,1           ;1
      BSF   STATO,0           ;1  23
      MOVLW D'63'             ;1          leggo cella per determinare
      CALL  EEREADW           ;9  33      se orologio o scritta

      DNOP                    ;35
      DNOP                    ;37

      BTFSC STATO,4           ;1    partenza con interruttore
      GOTO  STATO_4           ;2  40      di WR inserito

      BTFSC   STATO,3         ;1          verifico in che stato sono
      GOTO    STATO_3         ;2  42      vado velocit… E tono

      BTFSC   STATO,2         ;1          verifico in che stato sono
      GOTO    STATO_2         ;2  44      cambio scritta fissa

      BTFSC   STATO,1         ;1          verifico in che stato sono
      GOTO    STATO_1         ;2  46      cambio scritta scorrevole

      NOP                     ;1
      GOTO  STATO_0           ;2  48

;*********************************************************************
;     STATO 0    scelta X display    ora/scritta/alternanza
;*********************************************************************
STATO_4                       ;40   partenza con scritta
      CALL  RIT_6             ;6  46      inclinata causa WR

STATO_0                       ;48
      CALL  TASTO_P           ;14         Š tasto premuto
      SKIPSZ                  ;1
      GOTO    PULFIN          ;65   se no salto il set ora

      MOVF    PortB,W         ;1          verifico se pulsanti entrambi
      ANDLW   B'11000000'     ;1          premuti se si vado in STATME
      SKIPNZ                  ;1
      GOTO    STATO_9         ;2  69      se si salto a memorizzazione

      BTFSC EEData,6          ;1  69      se Š alternanza ora/scritta
      BCF   EEData,7          ;1          pongo scritta
      BTFSC EEData,7          ;1          cambio colore di fondo
      GOTO  STATO_7           ;2  73      in STATO8
      GOTO  STATO_8           ;2  74
```

```
;*********************************************************************
PULFIN                      ;65
      CLRF    STATO         ;
      CLRF    CNTCA         ;
      CLRF    PUNCA         ;68
      MOVLW D'17'           ;
      CALL  RIT_P2          ;58 126     ritardo per sincronizzare
      RETURN                ;2  128

;*********************************************************************
STATO_7                     ;73
      BTFSC   PortB,6       ;           set minuti se premuto
      INCF    MiU,1         ;
      BTFSC   PortB,7       ;           set ore se premuto
      INCF    HrU,1         ;
      CLRF    SeU           ;79
      MOVLW   D'82'         ;           tempo di attesa per
      BTFSC Count2,6        ;
      MOVWF   Count2        ;82   prossimo incremento
      MOVLW D'13'           ;           ritardo
      CALL  RIT_P0          ;44 126
      RETURN                ;2  128

;*********************************************************************
STATO_8                     ;74
      MOVLW D'62'           ;1          leggo cella per determinare
      CALL  EEREADW         ;9  84      il colore di fondo
      BTFSC   PortB,7       ;1
      SWAPF EEData,1        ;1
      BCF   EEData,3        ;1
      INCF  EEData,1        ;1
      BTFSC PortB,7         ;1
      SWAPF EEData,1        ;1   90
      CALL  MEMEEP          ;12  102

      MOVLW D'6'            ;
      CALL  RIT_P1          ;24 126
      GOTO    FINATT        ;2  128

;*********************************************************************
STATO_9                     ;69
      MOVLW   B'10000000'   ;1          scelta se visualizzare
      BCF   EEData,6        ;1          testo o orologio e azzero
      XORWF EEData,1        ;1          alternanza
      BTFSC Count2,6        ;1  73     verifico se e giro di OLD
      BSF   EEData,6        ;1
      CALL    MEMEEP        ;12 86      MEMORIZZA

      MOVLW D'11'           ;
      CALL  RIT_P2          ;40 126
      GOTO    FINATT        ;2  128
```

```
;*********************************************************************
;STATO_X   ( per stato 1-2 ) cambio scritta scorrevole e fissa
;*********************************************************************
STATO_2                     ;44
      MOVLW   PRIMFIS       ;1    carattere scritta fissa
      MOVWF   Ptrtxt        ;
STATO_1                     ;46
      CLRF    Count3        ;1         blocco la scritta scorrevole
      MOVLW D'8'            ;1
      BTFSS STATO,2         ;1    stato diverso da 2 PUNCA = 0
      MOVWF PUNCA           ;1  50
      CALL  TASTO_P         ;14 64      Š tasto premuto
      SKIPSZ                ;1
      GOTO    STAT1_A1      ;67   se no salto

      MOVF    PortB,W       ;
      ANDLW   B'11000000'   ;            pulsanti entrambi pigiati
      SKIPNZ                                  ;
      GOTO    STORCAR       ;71   se si salto a memorizzazione

      CALL  SCELTA_CA       ;20 90      rit con carattere in TABCA

STAT1_A2
      CALL CONVER           ;15 105

      MOVLW H'16'           ;1    preparo il puntatore alla
      ADDWF PUNCA,W         ;1  107     cella per display
      MOVWF   FSR           ;1    punto alla tabella display
      MOVF    TABCA,W       ;1    e metto il nuovo carattere
      MOVWF   INDF          ;1  110
      MOVLW D'3'            ;
      CALL  RIT_P2          ;16   ritardo
      RETURN                ;2  128

STAT1_A1                    ;
      MOVLW D'5'            ;
      CALL  RIT_P1          ;21
      GOTO  STAT1_A2        ;2  90

;*********************************************************************
;         Memorizzazione e cambio stato
;*********************************************************************
STORCAR                     ;71
      MOVF    Ptrtxt,W      ;1  72
      ADDWF   PUNCA,W       ;1         memorizzazione del puntatore
      MOVWF   EEAdr         ;1         tabella caratteri in eeprom
      XORLW D'61'           ;1  75     ultima cella di scrittura
      SKIPSZ                ;1  76     se non Š ultima cella
      INCF    Ptrtxt,1      ;1       incremento il puntatore Ptrtxt
      SKIPSZ                ;1         e il PUNCA
      INCF  PUNCA,1         ;1
      CLRF    CNTCA         ;1  80     azzero carattere
      MOVF    TABCA,W       ;1
      MOVWF   EEData        ;1
      XORLW   CarBL         ;1  83
      SKIPNZ                ;1    verifico se Š il BLOC se si
      CLRF  PUNCA           ;1    azzero PUNCA
      SKIPNZ                ;1
      RLF   STATO,1         ;1    cambio stato e non memorizzo
```

```
      SKIPSZ                  ;1   88
      CALL    MEMEEP          ;12 100

      MOVLW D'7'              ;           ritardo sincronizzazione riga
      CALL  RIT_P2            ;28 128

FINATT
      MOVLW D'224'            ;224  Ritardo per poter scrivere
      MOVWF CNT_0             ;           la EEPROM a bordo del PIC
      CALL  BLKLINE           ;           nessuna operazione su EEPROM
      GOTO  INCTIM            ;           per almeno 14 milli secondi

;*****************************************************************
;   STATO_3   cambio velocit… di scorrimento scritta
;*****************************************************************
STATO_3                       ;42
      MOVF  EEData,W          ;1
      MOVWF   CNTCA           ;1  44
      CALL  TASTO_P           ;14 58  Š tasto premuto
      SKIPNZ                  ;1
      GOTO  STAT3_61          ;61

      MOVLW D'20'             ;     ritardo
      CALL RIT_P1             ;66 126
      RETURN                  ;2   128

;---------------------------------------------------------------

STAT3_61                      ;    61
      BTFSS PortB,6           ;     verifico se devo variare velocit…
      GOTO  SPECIALE          ;    64     o se devo cambiare il tono

      CLRF  Count3            ;1  58      azzero il count3 per sommare
      RLF   CNTCA,1           ;1    correttamente
      BCF   CNTCA,0           ;1    vario la velocit… di scorrimento
      BTFSC CNTCA,3           ;1    e la pongo a pixel o a caratteri
      BSF   CNTCA,0           ;1
      MOVLW D'16'             ;1   69
      BTFSC CNTCA,3           ;1
      ADDWF CNTCA,1           ;1
      BCF   CNTCA,3           ;1   72
      BTFSC CNTCA,4           ;1
      BSF   CNTCA,3           ;1
      MOVLW B'11110000'       ;1   75
      ANDWF EEData,1          ;1
      MOVF    CNTCA,W         ;1
      ANDLW B'00001111'       ;1
      IORWF   EEData,1        ;1   79
      CALL    MEMEEP          ;12  91    MEMORIZZA
      MOVLW D'10'             ;
      CALL  RIT_P0            ;35 126
      GOTO  FINATT            ;2   128
```

```
;-------------------------------------------------------------

SPECIALE                      ;64
      MOVLW       B'00100000' ;1          scelta tono singolo/bitonale
      XORWF EEData,1          ;1
      CALL    MEMEEP          ;12 78      MEMORIZZA

      MOVLW D'14'             ;
      CALL  RIT_P1            ;48 126
      GOTO    FINATT          ;2   128

;---------------------------------------------------------------------

MEMEEP  BSF     Status,RP0    ;1
      BSF     EECON1,WREN     ;2
      MOVLW   055             ;3
      MOVWF   EECON2          ;4          MEMORIZZAZIONE EEPROM
      MOVLW   0AA             ;5
      MOVWF   EECON2          ;6
      BSF     EECON1,WR       ;7
      BCF     EECON1,WREN     ;2
      BCF     Status,RP0      ;8
      RETURN                  ;10

;*********************************************************************
;     Scelta del carattere in modo circolare e ....
;        ....
;*********************************************************************
SCELTA_CA
      BTFSC   PortB,7         ;1
      INCF  CNTCA,1           ;1    incremento carattere
      BTFSC   PortB,6         ;1
      DECF  CNTCA,1           ;1  4       decremento carattere
      MOVF    CNTCA,W         ;1
      XORLW   D'255'          ;1
      MOVLW D'39'             ;1
      SKIPNZ                  ;1
      MOVWF CNTCA             ;1  9
      MOVF    CNTCA,W         ;1
      XORLW   D'40'           ;1          verifico se fondo scala
      SKIPNZ                  ;1          o inizio
      CLRF  CNTCA             ;1  13
      MOVLW   D'83'           ;1  14      velocit… di cambio carattere
      BTFSC Count2,6          ;1
      MOVWF   Count2          ;1  16
      RETURN                  ;2  18
```

```
;*********************************************************************
;   CONVER  converto il numero nel puntatore alla tabella
;*********************************************************************
CONVER  MOVLW   HIGH Texte    ;1         conversione del numero CNTCA
      MOVWF   PClath          ;2         al puntatore del carattere
      MOVF    CNTCA,W         ;3
      CALL    Texte           ;(6) 9

      MOVWF   TABCA           ;10
      CLRF    PClath          ;11
      RETURN                  ;13

;*********************************************************************
;     Sistema la flag se il tasto Š premuto e ha superato la
;       soglia di attivazione normale e old  ( incremento ogni 20mS )
;*********************************************************************
TASTO_P
      COMF    PortB,w         ;1
      ANDLW   B'11000000'     ;1         vedo se Š premuto un dei due
      MOVLW D'42'             ;1
      SKIPNZ                  ;1         o entrambi i tasti se si skip
      MOVWF   Count2          ;1         non premuti faccio clear
      INCF    Count2,1        ;1  6      contatore di quadri e increm
      MOVLW D'50'             ;1    tempo prima di servire la pigiata
      BTFSC Count2,6          ;1    testo se 1 vota o seconda
      MOVLW D'90'             ;1tempo per considerare old la pigiata
      XORWF   Count2,W        ;1         verifico se count e arrivato
      RETURN                  ;2  12

;*****************************************************
;    Incremento orologio
;*****************************************************
INCTIME CALL      DSINCK      ;16  18
      INCF    SubSec,1        ;1    Incremento 1/50 sec
      MOVLW   -D'50'          ;1   20
      ADDWF   SubSec,W        ;1
      SKIPNC                  ;1
      CLRF    SubSec          ;1
      SKIPNC                  ;1
      INCF    SeU,1           ;1     And increment Second Units
      SKIPNC                  ;1
      NOP                     ;1     seconds counter modulo 256
      MOVLW   -D'60'          ;1
      ADDWF   SeU,W           ;1 Carry if needed Second Units->Tens
      SKIPNC                  ;1  30
      CLRF    SeU             ;1
      SKIPNC                  ;1
INCF    MiU,1                 ;1    unit… di minuti
      MOVLW   -D'10'          ;1
      ADDWF   MiU,W           ;1  35
      SKIPNC                  ;1
      CLRF    MiU             ;1
      SKIPNC                  ;1
      INCF    MiT,1           ;1  39     decine di minuti
      MOVLW   -D'6'           ;1  40
      ADDWF   MiT,W           ;1
```

```
      SKIPNC                  ;1
      CLRF    MiT             ;1
      SKIPNC                  ;1
      INCF    HrU,1           ;1  45
      MOVLW   -D'10'          ;1
      ADDWF   HrU,W           ;1
      SKIPNC                  ;1
      CLRF    HrU             ;1
      SKIPNC                  ;1  50
      INCF    HrT,1           ;1
      MOVF    HrU,W           ;1    Now check for Hours=3D24
      BTFSC   HrT,0           ;1
      ADDLW   D'10'           ;1  54
      BTFSC   HrT,1           ;1
      ADDLW   -D'4'           ;1
      SKIPNC                  ;1
CLRF    HrU                   ;1  58   clear unita di ore
      SKIPNC                  ;1
      CLRF    HrT             ;1  60      clear decine di ore
      MOVLW D'63'             ;1
      CALL  EEREADW           ;9  70

      MOVLW CELLSP            ;
      MOVWF EEAdr             ;
      MOVLW B'00000011'       ;1
      BTFSC SeU,0             ;1
      BTFSC EEData,5          ;1
      MOVLW B'00000100'       ;1 76
      BSF     Status,RP0      ;1    select bank 1
      MOVWF       Roption     ;1    prescaler TMR0 / 2
      BCF     Status,RP0      ;1  79      select bank 0
      MOVLW D'14'             ;     ritardo
      CALL  RIT_P0            ;47  126
      RETURN                  ;2    128

;*****************************************************
;     prepara le celle da displeiare
;*****************************************************
PREREAD CALL      DSINCK      ;18
      MOVLW   D'63'           ;1
      CALL    EEREADW         ;9  28   estraggo da EEPROM la cella
                              ;          per la velocit… o no ????
      MOVWF DISPE             ;1         la metto in DISPE
      ANDLW B'00000111'       ;1
      ADDWF Count3,1          ;1         e la sommo a Count3
      MOVF  Count3,W          ;1
      ADDLW   -D'56'          ;*1  33numero di quadri (20mS quadro)
      SKIPNC                  ;*1-2 per velocit… scorrimento
      CLRF  Count3            ;*    36    scritta al primo ingresso
      SKIPNC                  ;*    partenza per arrivare a ...
      INCF    Ptrtxt,1        ;*1  37
      DNOP                    ;*    39
PREREA_39
      MOVF  Ptrtxt,W          ;1    Š puntatore all'ultimo carattere
      ADDLW -D'54'            ;1    se si azzero puntatore
      MOVLW -D'8'             ;1
      SKIPNZ                  ;1
      MOVWF Ptrtxt            ;1  44
      MOVF    Ptrtxt,W        ;1  45
```

```
      CALL  EEREADW           ;9  54
      XORLW   CarFI           ;1    verifico se Š ultimo carattere
      MOVLW -D'8'             ;1          del messaggio
      SKIPNZ                  ;1    se no salto .. altrimenti
      MOVWF   Ptrtxt          ;1  58      pongo puntatore a -7
      MOVF    Ptrtxt,W        ;1  59
PREREA_59
      MOVWF   EEAdr           ;1          metto puntatore carattere in
      MOVLW   D'22'           ;1    puntatore al segmento
      MOVWF   FSR             ;1  62      inizio tabella CA
PREREA_62
      MOVLW D'20'             ;
      CALL  RIT_P1            ;66 128

;-------------------------------------------------- NUOVA LINEA
      CALL  LETT_5                ;
;-------------------------------------------------- NUOVA LINEA
      GOTO  LETT_5                ;
;-----------------------------------------
;-----------------------------------------

PREH_34
      MOVLW D'5'              ;
      CALL  RIT_P2            ;22 56
      MOVLW ULTICAR           ;1
      GOTO  PREREA_59         ;2  59

;************************************************************
;     Prepara riga per display ore o scritta
;************************************************************
PREPH   CALL      DSINCK      ;18
      MOVLW   D'63'           ;1
      CALL    EEREADW         ;9  28      verifico se display ore
      BTFSC EEData,6          ;1    0=fissa/ora  1=alterna fissa/ora
      GOTO    PREH_TIME       ;31         se 1 vado a ...
      BTFSC EEData,7          ;1  310=scritta fissa se 1= ore salto
      GOTO    PREH_ORA        ;   33      se scritta vado a...
      GOTO  PREH_34           ;   34

PREH_TIME                     ;
      BTFSC SeU,2             ;  32
      GOTO  PREH_34           ;  34

PREH_ORA                      ;
      MOVLW   CarSP           ;1  34      Metto lo space in tutte
      MOVWF   CA0             ;1    le celle da  displeiare
      MOVWF CA1               ;1
      MOVWF CA2               ;1
      MOVWF CA8               ;1  38
      BCF     Status,C        ;1
      MOVF    HrT,W           ;1    Decine di ore
      CALL  MULT7             ;10
      MOVWF CA3               ;1  51
      MOVF    HrU,W           ;1    unit… di ore
      CALL  MULT7             ;10
      MOVWF CA4               ;1  63
      MOVLW   CarDP           ;1
      BTFSC   SeU,0           ;1    visualizzo i due punti o
      MOVLW   CarSP           ;1    lo space ogni secondo
```

```
      MOVWF   CA5             ;1  67
      MOVF    MiT,W           ;1          Decine di minuti
      CALL  MULT7             ;10
      MOVWF CA6               ;1  89
      MOVF    MiU,W           ;1   Decine di ore
      CALL  MULT7             ;10
      MOVWF CA7               ;1   91
      MOVLW D'10'             ;            ritardo
      CALL  RIT_P2            ;37  128

      MOVLW D'2'              ;     chiamo due righe nere
      MOVWF CNT_0             ;     e ritorner…
      GOTO  BLKLINE           ;

;*****************************************************
;  routines di moltiplicazione per sette
;*****************************************************
MULT7 MOVWF   TIME            ;1
      RLF     TIME,1          ;1   moltiplico x sette
      RLF     TIME,1          ;1         il valore in W e lo
      ADDWF   TIME,1          ;1         ritorno in W
      ADDWF   TIME,1          ;1
      ADDWF   TIME,W          ;1
      RETURN                  ;2 (8)

;*****************************************************
;     lettura dato in eeprom
;*****************************************************
LETT_5
      CALL  DSINCK            ;16 18
      CALL    EEREAD          ;20 38     leggo  carattere
      CALL    EEREAD          ;20 58     leggo  carattere
      CALL    EEREAD          ;20 78     leggo  carattere
      CALL    EEREAD          ;20 98     leggo  carattere
      CALL    EEREAD          ;20 118    leggo  carattere
      CALL    RIT_8           ;8  126    leggo  carattere
      RETURN                  ; 128

;-----------------------------------------------------

EEREAD
      CALL  LETTU             ;8    lettura della eeprom

      BTFSC EEAdr,6           ;ALTERNATIVO A PUNTARE ALLA
                              ;CELLA
      MOVLW   CarSP           ;ZERO DELLA EEPROM CON SPACE
      MOVWF   INDF            ;          metto dato a indirizzo
      INCF    FSR,1           ;          puntato da FSR
      XORLW   CarFI           ;          verifico se fine caratteri
      SKIPNZ                  ;          se si
      BSF   EEAdr,6           ;       setto di leggere sempre cella 0
      INCF    EEAdr,1         ;
      RETURN                  ;18

EEREADW
      MOVWF   EEAdr           ;1         lettura della eeprom
```

```
LETTU
      BSF     Status,RP0      ;2          alla cella puntata
      BSF     EEData,RD       ;3          dal W prima della chiamata
      BCF     Status,RP0      ;4
      MOVF    EEData,W        ;5          ritorno dato in accumulatore
      RETURN                  ;7

;*****************************************************
;     tabella di riferimento caratteri asci
;*****************************************************
Texte   ADDWF   PCL,F
DTexte  equ     $

      RETLW   CarBL           ; -
      RETLW   CarA            ; 0
      RETLW   CarB            ; 1
      RETLW   CarC            ; 2
      RETLW   CarD            ; 3
      RETLW   CarE            ; 4
      RETLW   CarF            ; 5
      RETLW   CarG            ; 6
      RETLW   CarH            ; 7
      RETLW   CarI            ; 8
      RETLW   CarJ            ; 9
      RETLW   CarK            ; 10
      RETLW   CarL            ; 11
      RETLW   CarM            ; 12
      RETLW   CarN            ; 13
      RETLW   CarO            ; 14
      RETLW   CarP            ; 15
      RETLW   CarQ            ; 16
      RETLW   CarR            ; 17
      RETLW   CarS            ; 18
      RETLW   CarT            ; 19
      RETLW   CarU            ; 20
      RETLW   CarV            ; 21
      RETLW   CarW            ; 22
      RETLW   CarX            ; 23
      RETLW   CarY            ; 24
      RETLW   CarZ            ; 25
      RETLW   CarSP           ; 26
      RETLW   Car0            ; 27
      RETLW   Car1            ; 28
      RETLW   Car2            ; 29
      RETLW   Car3            ; 30
      RETLW   Car4            ; 31
      RETLW   Car5            ; 32
      RETLW   Car6            ; 33
      RETLW   Car7            ; 34
      RETLW   Car8            ; 35
      RETLW   Car9            ; 36
      RETLW   CarDP           ; 37
      RETLW   CarFI           ; 38
      END
```

3.3.2. GENERADOR DE VÍDEO – AFICIONADO Nº2.

En este caso el generador dispone de varios patrones de prueba:

1) Barras de color.
2) Raster.
3) Cross-hatch.
4) Puntos.

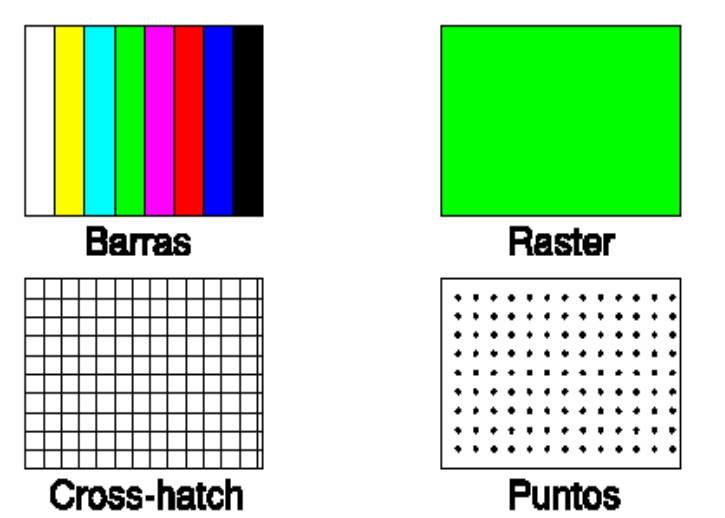


Figura 3.8 – Generador de vídeo-aficionado nº2.

Todos los patrones son a pantalla completa, siendo el barrido entrelazado en los modos Barras y Raster, y no entrelazado en los modos Cross-hatch y Puntos.

Del mismo modo que en el caso anterior, el generador está constituido principalmente por dos circuitos integrados:

1) Un microcontrolador PIC16F84, encargado de la generación de las señales de vídeo (sincronismo compuesto, R, G, y B), y el control del teclado.
2) Un conversor de vídeo MC1377, que genera la señal de vídeo-compuesto.

El teclado consta de dos conmutadores, mediante los cuales se pueden seleccionar los diferentes modos de funcionamiento.

También se dispone de mandos adicionales que permiten eliminar las componentes de color, la señal de luminancia, la señal de crominancia, y la salva de color.

El circuito proporciona una salida de video compuesto de aproximadamente 1Vp-p en carga de 75 Ω.

La alimentación del circuito es de 12VDC. El conversor de vídeo se alimenta directamente a través de un diodo de protección junto con unos condensadores de filtrado, mientras que el microcontrolador se alimenta con 5VDC a través de un regulador de tensión integrado del tipo 7805.

En la figura 3.9 se muestra el esquema del circuito, donde pueden verse las aclaraciones anteriores.

Para la construcción del prototipo se tuvo que redibujar el circuito y diseñar la placa de circuito impreso, ya que el autor del proyecto no la suministraba. La placa está diseñada con el programa de diseño electrónico "Protel 98". Se encuentra realizada a una cara, y todos los procesos de elaboración de la misma se comentaran más adelante en el capítulo 6.

La placa de circuito impreso, la situación de los componentes, y la lista de componentes se muestran en las figuras 3.10, 3.11, y 3.12 respectivamente.

El software de la aplicación está escrito en lenguaje ensamblador con un editor de texto estándar, y se ensambló con la herramienta de Microchip "MPASM for Windows". El código fuente completo de la aplicación se suministra al final del capitulo.

Para cargar la memoria eeprom del microcontrolador se utilizó el programa "ICProg", junto con el programador "TE-20".

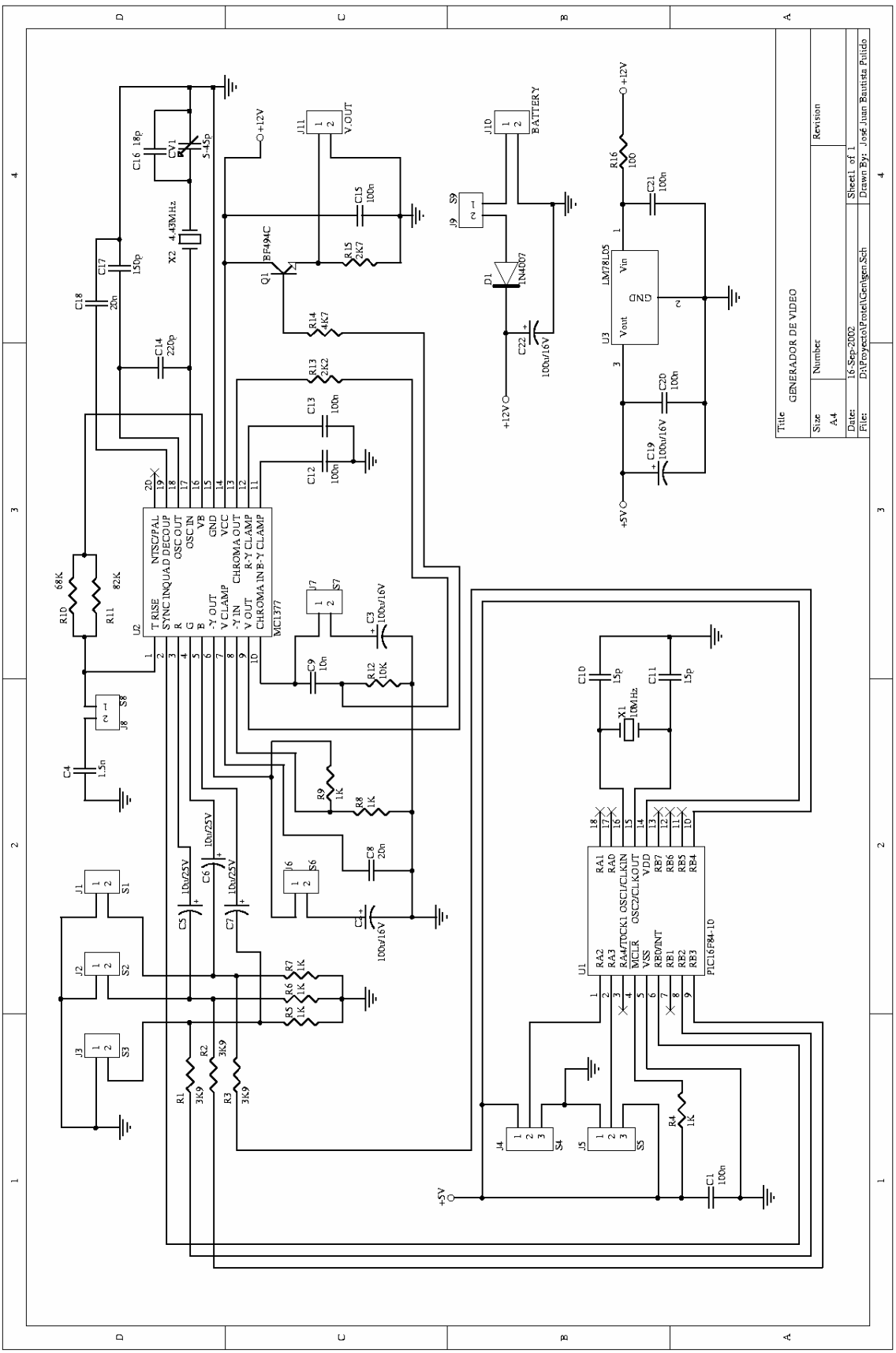


Figura 3.9 – Esquema del generador de vídeo-aficionado nº2.

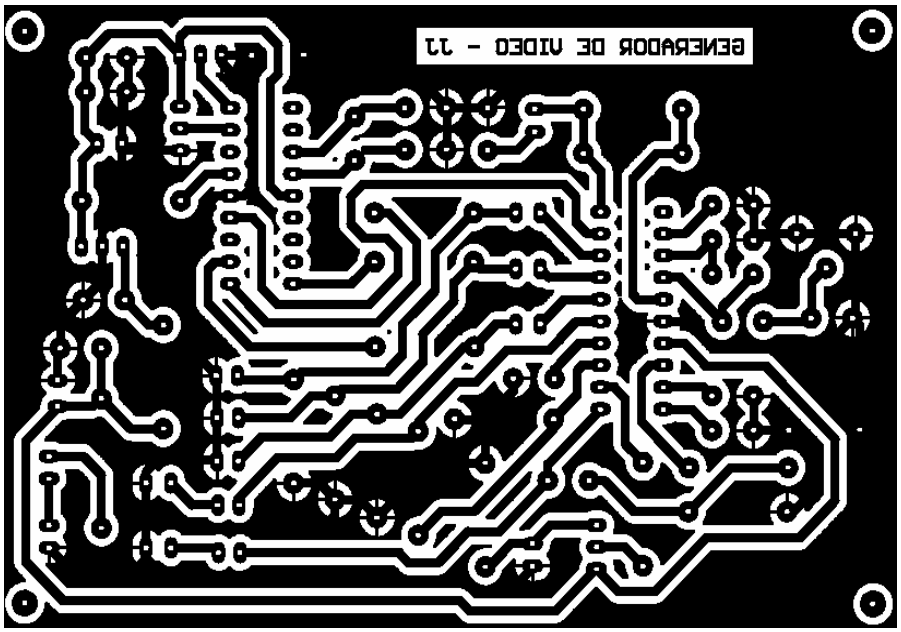


Figura 3.10 – PCB del generador de vídeo-aficionado nº2.

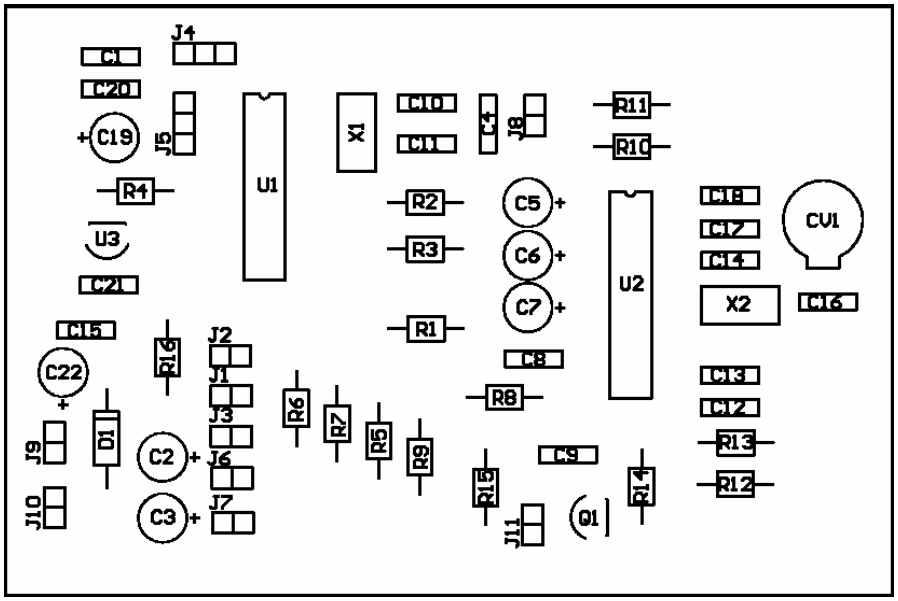


Figura 3.11 – Componentes del generador de vídeo-aficionado nº2.

R01 3K9	C13 0.1µ
R02 3K9	C14 220p
R03 3K9	C15 0.1µ
R04 1K	C16 18p
R05 1K	C17 150p
R06 1K	C18 .02µ
R07 1K	C19 100µ/16V
R08 1K	C20 0.1µ
R09 1K	C21 0.1µ
R10 68K	C22 100µ/16V
R11 82K	CV1 TRIMMER 5-45p
R12 10K	D1 1N4007
R13 2K2	Q1 BF494C
R14 4K7	U1 PIC16F84-10
R15 2K7	U2 MC1377
R16 100	U3 LM78L05
C01 0.1µ	X1 10.000MHz
C02 100µ/16V	X2 4.4336MHz
C03 100µ/16V	S1 LLAVE 2 POSICIONES
C04 1500p	S2 LLAVE 2 POSICIONES
C05 10µ/25V	S3 LLAVE 2 POSICIONES
C06 10µ/25V	S4 LLAVE 2 POSICIONES
C07 10µ/25V	S5 LLAVE 2 POSICIONES
C08 .02µ	S6 LLAVE 2 POSICIONES
C09 .01µ	S7 LLAVE 2 POSICIONES
C10 15p	S8 LLAVE 2 POSICIONES
C11 15p	S9 LLAVE 2 POSICIONES
C12 0.1µ	

Figura 3.12 – Lista de componentes del generador de vídeo-aficionado nº2.

A continuación se muestra el código fuente de la aplicación:

```
; ***** GENERADOR DE PATRONES PARA VIDEO *****
; VERSION 2.01
; GEN201.ASM
; (C) M. MAGGI - 30/08/1997

        list   p=16f84

;DEFINICION DE PUERTOS:
;PORTB(0): SYNC
;PORTB(2): AZUL
;PORTB(3): ROJO
;PORTB(4): VERDE
;NO USAR EL BIT 1 DEL PORTB
;PARA LOS PATRONES DE RASTER Y BARRAS EL VIDEO ES ENTRELAZADO
;LOS PUNTOS Y EL CROSSHATCH SE HACEN CON VIDEO NO ENTRELAZADO PARA
;EVITAR EL "FLICKER"

CBLOCK  0X0C ;VARIABLES
      DURHOR,CANTHB1,CANTHB2,BLKLIN,CANTPRE,DUREQU,CANTVER,DURVER,
      CANTPOS,TIEMPO,FIELD,CARRY,WHITE,YELLOW,CYAN,GREEN,MAGEN,RED,
      BLUE,BLACK,CANTLIN
ENDC

PORTA   EQU    5
TRISA   EQU    85H
PORTB   EQU    6
TRISB   EQU    86H
STATUS  EQU    3
RP0     EQU    5
BLANCO  EQU    B'00011101'
AMARIL  EQU    B'00011001'
CYANO   EQU    B'00010101'
VERDE   EQU    B'00010001'
MAGENT  EQU    B'00001101'
ROJO    EQU    B'00001001'
AZUL    EQU    B'00000101'
NEGRO   EQU    B'00000001'

        CLRF   PORTA ;TODOS LOS BITS EN 0
        CLRF   PORTB ;TODOS LOS BITS EN 0
        BSF    STATUS,RP0 ;SELECCIONA BANCO DE REGISTROS 1
        MOVLW  B'11111111'
        MOVWF  TRISA ;TODOS LOS BITS DEL PUERTO A COMO ENTRADAS
        CLRF   TRISB^80H ;TODOS LOS BITS DEL PUERTO B COMO SALIDA
        BCF    STATUS,RP0 ;SELECCIONA BANCO DE REGISTROS O

        MOVLW  0
        MOVWF  CARRY ;VARIABLE CONTROLAR EL ESTADO DEL CARRY
        RRF    CARRY ;CARRY FLAG A "0"
        MOVLW  B'10101010'
        MOVWF  FIELD ;CONTROL DEL CAMPO

LECTURA BTFSS  PORTA,3 ;SE LEE EL TECLADO
        GOTO   LECT1 ;SE USAN LOS BITS 2 Y 3 DEL PUERTO A
        BTFSC  PORTA,2 ;FUNCION: BIT3 BIT2
        GOTO   INICIO3 ;BARRAS 0 0
        GOTO   INICIO2 ;RASTER 0 1
        LECT1  BTFSC PORTA,2 ;CROSSHATCH 1 0
        GOTO   INICIO1 ;PUNTOS 1 1
```

```
;***** BARRAS DE COLOR *****

INICIO  RRF    FIELD ;CARRY PASA AL BIT 7 DE FIELD, BIT 0 AL CARRY
        MOVLW  D'3' ;LINEAS SIN VIDEO LUEGO DE LA POSECUALIZACION
        BTFSS  FIELD,0 ;SI ES EL CAMPO 1 SE HACEN SOLO 3 LINEAS
        MOVLW  D'4' ;4 LINEAS EN EL CAMPO 2
        MOVWF  BLKLIN
        MOVLW  D'99'
        MOVWF  CANTHB1 ;CANTIDAD DE LINEAS HORIZONTALES EN UN BLOQUE
        MOVLW  D'3'
        MOVWF  CANTHB2 ;CANTIDAD DE BLOQUES (3)
        MOVLW  5
        MOVWF  CANTPRE ;PULSOS DE PREECUALIZACION
        MOVLW  5
        MOVWF  CANTVER ;PULSOS DE SINCRONISMO VERTICAL
        MOVLW  5
        MOVWF  CANTPOS ;PULSOS DE POSECUALIZACION
PREEQU  BCF    PORTB,0 ;DURACION: 2,6µS ABAJO
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
LOOP1   DECFSZ DUREQU ;SE COMPLETAN LOS 32µS ARRIBA
        GOTO   LOOP1
        NOP
        NOP
        DECFSZ CANTPRE
        GOTO   PREEQU
        NOP
VERT    BCF    PORTB,0
        MOVLW  D'22'
        MOVWF  DURVER
LOOP2   DECFSZ DURVER
        GOTO   LOOP2
        BSF    PORTB,0 ;DURACION: 4.8µS ARRIBA ("SERRATED PULSES")
        MOVLW  2
        MOVWF  TIEMPO
TIME    DECFSZ TIEMPO
        GOTO   TIME
        NOP
        DECFSZ CANTVER
        GOTO   VERT
        NOP
POSEQU  BCF    PORTB,0
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
LOOP3   DECFSZ DUREQU
        GOTO   LOOP3
        NOP
        NOP
        DECFSZ CANTPOS
        GOTO   POSEQU
        NOP
```

```
;SE EMPIEZAN A BARRER LAS LINEAS HORIZONTALES
;LA PRIMERA LINEA ES COMPLETA EN EL CAMPO 1, EN TANTO QUE ES SOLO
;MEDIA LINEA
;EN EL CAMPO 2, Y NO COMIENZA CON UN PULSO DE SINCRONISMO

        RLF    PORTB ;1 O 1/2 LINEA H SEGUN EL CAMPO
        NOP          ;SE PASA EL CARRY AL BIT 0 DEL PUERTO B
        NOP          ;CAMPO 1: 1 LINEA Y PULSO DE SINC (CARRY=0)
        NOP          ;CAMPO 2: 1/2 LINEA SIN PULSO DE SINC (CARRY=1)
        NOP
        NOP
        NOP
        NOP
        MOVLW  D'21' ;TIEMPO PARA 1/2 H (80 CICLOS TOTAL)
        BTFSS  PORTB,0 ;SI HAY H SYNC (CAMPO 1) SE AGREGA MAS TIEMPO
        ADDLW  D'27' ;TIEMPO PARA 1 H (160 CICLOS TOTAL)
        MOVWF  DURHOR
        BSF    PORTB,0
        BTFSS  FIELD,0
        GOTO   NEXT ;SE PIERDE 1 CICLO MAS (SOLO 1/2 H)
NEXT    BCF    PORTB,1
        NOP
LOOP    DECFSZ DURHOR
        GOTO   LOOP
        NOP

;SE HACEN 3 O 4 LINEAS EN BLANCO PARA CUMPLIR CON LAS 625 LINEAS DE LA
;NORMA N
;SI ES EL CAMPO 1 SE HACEN SOLO 3 LINEAS, YA QUE ANTES SE HIZO 1 DE
;MAS

HORIZ   BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
TIME3   DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   TIME3
        NOP
        NOP
        MOVLW  D'48'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
LOOPH3  DECFSZ DURHOR
        GOTO   LOOPH3
        NOP
        DECFSZ BLKLIN
        GOTO   HORIZ
        NOP

;SE HACEN 3 BLOQUES DE 99 LINEAS HORIZONTALES
;3*(99+1)=300 LINEAS

HORIZ1  BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
```

```
TIME1   DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   TIME1
        NOP
        NOP
        MOVLW  D'31'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        MOVLW  5
        MOVWF  WHITE
        MOVWF  YELLOW
        MOVWF  CYAN
        MOVWF  GREEN
        MOVWF  MAGEN
        MOVWF  RED
        MOVWF  BLUE
        MOVWF  BLACK
        MOVLW  BLANCO
        MOVWF  PORTB
WHITE1  DECFSZ WHITE
        GOTO   WHITE1
        MOVLW  AMARIL
        MOVWF  PORTB
YELLO1  DECFSZ YELLOW
        GOTO   YELLO1
        MOVLW  CYANO
        MOVWF  PORTB
CYAN1   DECFSZ CYAN
        GOTO   CYAN1
        MOVLW  VERDE
        MOVWF  PORTB
GREEN1  DECFSZ GREEN
        GOTO   GREEN1
        MOVLW  MAGENT
        MOVWF  PORTB
MAGEN1  DECFSZ MAGEN
        GOTO   MAGEN1
        MOVLW  ROJO
        MOVWF  PORTB
RED1    DECFSZ RED
        GOTO   RED1
        MOVLW  AZUL
        MOVWF  PORTB
BLUE1   DECFSZ BLUE
        GOTO   BLUE1
        MOVLW  NEGRO
        MOVWF  PORTB
BLACK1  DECFSZ BLACK
        GOTO   BLACK1
        NOP
        NOP
        NOP
        NOP
        DECFSZ CANTHB1
        GOTO   HORIZ1
        NOP
```

```
HORIZ2  BCF    PORTB,0 ;BIT 0 BAJO
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
TIME2   DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   TIME2
        MOVLW  D'99'
        MOVWF  CANTHB1
        MOVLW  D'31'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        MOVLW  5
        MOVWF  WHITE
        MOVWF  YELLOW
        MOVWF  CYAN
        MOVWF  GREEN
        MOVWF  MAGEN
        MOVWF  RED
        MOVWF  BLUE
        MOVWF  BLACK
        MOVLW  BLANCO
        MOVWF  PORTB
WHITE2  DECFSZ WHITE
        GOTO   WHITE2
        MOVLW  AMARIL
        MOVWF  PORTB
YELLO2  DECFSZ YELLOW
        GOTO   YELLO2
        MOVLW  CYANO
        MOVWF  PORTB
CYAN2   DECFSZ CYAN
        GOTO   CYAN2
        MOVLW  VERDE
        MOVWF  PORTB
GREEN2  DECFSZ GREEN
        GOTO   GREEN2
        MOVLW  MAGENT
        MOVWF  PORTB
MAGEN2  DECFSZ MAGEN
        GOTO   MAGEN2
        MOVLW  ROJO
        MOVWF  PORTB
RED2    DECFSZ RED
        GOTO   RED2
        MOVLW  AZUL
        MOVWF  PORTB
BLUE2   DECFSZ BLUE
        GOTO   BLUE2
        MOVLW  NEGRO
        MOVWF  PORTB
```

```
BLACK2  DECFSZ BLACK
        GOTO   BLACK2
        NOP
        NOP
        NOP
        NOP
        DECFSZ CANTHB2
        GOTO   HORIZ1
        NOP

;ESTA ULTIMA LINEA/MEDIA LINEA, LA 305, LA USO PARA CARGAR VARIABLES

        BCF    PORTB,0 ;BIT 0 PASA A NIVEL BAJO
        NOP ;PIERDO TIEMPO PARA
        NOP ;HACER LOS 4,8µS
        MOVLW  0 ;NO USAR EL BIT 1 DEL PORTB, BIT 0 = SYNC
        BTFSC  FIELD,0
        MOVLW  1
        MOVWF  CARRY
        NOP
        MOVLW  D'15'
        BTFSC  FIELD,0
        ADDLW  D'24'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 PASA A NIVEL ALTO
        BTFSS  FIELD,0
        GOTO   NEXT1
NEXT1   NOP
        NOP
LOOPH5  DECFSZ DURHOR
        GOTO   LOOPH5
        RRF    CARRY ;CARRY = 1 SI 1 H, CARRY = 0 SI 1/2 H
        BTFSC  PORTA,2
        GOTO   LECTURA
        BTFSC  PORTA,3
        GOTO   LECTURA
        NOP
        NOP
        NOP
        NOP
        NOP
        GOTO   INICIO

;***** RASTER *****

INICIO1 RRF    FIELD
        MOVLW  D'3'
        BTFSS  FIELD,0
        MOVLW  D'4'
        MOVWF  BLKLIN
        MOVLW  D'99'
        MOVWF  CANTHB1
        MOVLW  D'3'
        MOVWF  CANTHB2
        MOVLW  5
        MOVWF  CANTPRE
        MOVLW  5
        MOVWF  CANTVER
        MOVLW  5
        MOVWF  CANTPOS
```

```
APREEQU BCF    PORTB,0 ;DURACION: 2,6µS ABAJO
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
ALOOP1  DECFSZ DUREQU ;SE COMPLETAN LOS 32µS ARRIBA
        GOTO   ALOOP1
        NOP
        NOP
        DECFSZ CANTPRE
        GOTO   APREEQU
        NOP
AVERT   BCF    PORTB,0
        MOVLW  D'22'
        MOVWF  DURVER
ALOOP2  DECFSZ DURVER
        GOTO   ALOOP2
        BSF    PORTB,0 ;DURACION: 4.8µS ARRIBA ("SERRATED PULSES")
        MOVLW  2
        MOVWF  TIEMPO
ATIME   DECFSZ TIEMPO
        GOTO   ATIME
        NOP
        DECFSZ CANTVER
        GOTO   AVERT
        NOP
APOSEQU BCF    PORTB,0
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
ALOOP3  DECFSZ DUREQU
        GOTO   ALOOP3
        NOP
        NOP
        DECFSZ CANTPOS
        GOTO   APOSEQU
        NOP
        RLF    PORTB ;1 O 1/2 LINEA H SEGUN EL CAMPO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  D'21' ;TIEMPO PARA 1/2 H (80 CICLOS TOTAL)
        BTFSS  PORTB,0
        ADDLW  D'27' ;TIEMPO PARA 1 H (160 CICLOS TOTAL)
        MOVWF  DURHOR
        BSF    PORTB,0
        BTFSS  FIELD,0
        GOTO   ANEXT ;SE PIERDE 1 CICLO MAS (SOLO 1/2 H)
```

```
ANEXT   BCF    PORTB,1
        NOP
ALOOP   DECFSZ DURHOR
        GOTO   ALOOP
        NOP
AHORIZ  BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
ATIME3  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   ATIME3
        NOP
        NOP
        MOVLW  D'48'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
ALOOPH3 DECFSZ DURHOR
        GOTO   ALOOPH3
        NOP
        DECFSZ BLKLIN
        GOTO   AHORIZ
        NOP
AHORIZ1 BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
ATIME1  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   ATIME1
        NOP
        NOP
        MOVLW  D'44'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  B'00011101'
        MOVWF  PORTB
ALOOPH4 DECFSZ DURHOR
        GOTO   ALOOPH4
        MOVLW  B'00000001'
        MOVWF  PORTB
        DECFSZ CANTHB1
        GOTO   AHORIZ1
        NOP
        BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
ATIME2  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   ATIME2
        MOVLW  D'99'
        MOVWF  CANTHB1
        MOVLW  D'44'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
```

```
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  B'00011101'
        MOVWF  PORTB
ALOOPH5 DECFSZ DURHOR
        GOTO   ALOOPH5
        MOVLW  B'00000001'
        MOVWF  PORTB
        DECFSZ CANTHB2
        GOTO   AHORIZ1
        NOP

;ESTA ULTIMA LINEA/MEDIA LINEA, LA 305, LA USO PARA CARGAR VARIABLES

        BCF    PORTB,0 ;BIT 0 PASA A NIVEL BAJO
        NOP ;PIERDO TIEMPO PARA
        NOP ;HACER LOS 4,8µS
        MOVLW  0 ;NO USAR EL BIT 1 DEL PORTB, BIT 0 = SYNC
        BTFSC  FIELD,0
        MOVLW  1
        MOVWF  CARRY
        NOP
        MOVLW  D'15'
        BTFSC  FIELD,0
        ADDLW  D'24'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 PASA A NIVEL ALTO
        BTFSS  FIELD,0
        GOTO   ANEXT1
ANEXT1  NOP
        NOP
ALOOPH6 DECFSZ DURHOR
        GOTO   ALOOPH6
        RRF    CARRY ;CARRY = 1 SI 1 H, CARRY = 0 SI 1/2 H
        BTFSS  PORTA,2
        GOTO   LECTURA
        BTFSC  PORTA,3
        GOTO   LECTURA
        NOP
        NOP
        NOP
        NOP
        NOP
        GOTO   INICIO1
```

```
;***** CROSSHATCH *****

INICIO2 RRF    FIELD
        NOP
        NOP
        MOVLW  D'4'
        MOVWF  BLKLIN
        MOVLW  D'28'
        MOVWF  CANTHB1
        MOVLW  D'10'
        MOVWF  CANTHB2
        MOVLW  4
        MOVWF  CANTPRE ;SOLO 4 PULSOS POR SER VIDEO NO ENTRELAZADO
        MOVLW  5
        MOVWF  CANTVER
        MOVLW  5
        MOVWF  CANTPOS
BPREEQU BCF    PORTB,0 ;DURACION: 2,6µS ABAJO
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
BLOOP1  DECFSZ DUREQU ;SE COMPLETAN LOS 32µS ARRIBA
        GOTO   BLOOP1
        NOP
        NOP
        DECFSZ CANTPRE
        GOTO   BPREEQU
        NOP
BVERT   BCF    PORTB,0
        MOVLW  D'22'
        MOVWF  DURVER
BLOOP2  DECFSZ DURVER
        GOTO   BLOOP2
        BSF    PORTB,0 ;DURACION: 4.8µS ARRIBA ("SERRATED PULSES")
        MOVLW  2
        MOVWF  TIEMPO
BTIME   DECFSZ TIEMPO
        GOTO   BTIME
        NOP
        DECFSZ CANTVER
        GOTO   BVERT
        NOP
BPOSEQU BCF    PORTB,0
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
BLOOP3  DECFSZ DUREQU
        GOTO   BLOOP3
        NOP
        NOP
        DECFSZ CANTPOS
        GOTO   BPOSEQU
```

```
        NOP
        NOP ;1/2 LINEA H (NO ENTRELAZADO)
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  D'21' ;TIEMPO PARA 1/2 H (80 CICLOS TOTAL)
        NOP
        NOP
        MOVWF  DURHOR
        NOP
        NOP
        NOP
        NOP
        BCF    PORTB,1
        NOP
BLOOP   DECFSZ DURHOR
        GOTO   BLOOP
        NOP
BHORIZ  BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
BTIME3  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   BTIME3
        NOP
        NOP
        MOVLW  D'48'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
BLOOPH3 DECFSZ DURHOR
        GOTO   BLOOPH3
        NOP
        DECFSZ BLKLIN
        GOTO   BHORIZ
        NOP
BHORIZ1 BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
BTIME1  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   BTIME1
        NOP
        NOP
        MOVLW  9
        MOVWF  CANTLIN
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
```

```
BLOOPHA MOVLW  B'00011100'
        ADDWF  PORTB
        SUBWF  PORTB
        NOP
        MOVLW  2
        MOVWF  DURHOR
BLOOPH4 DECFSZ DURHOR
        GOTO   BLOOPH4
        DECFSZ CANTLIN
        GOTO   BLOOPHA
        NOP
        MOVLW  B'00011100'
        ADDWF  PORTB
        SUBWF  PORTB
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        DECFSZ CANTHB1
        GOTO   BHORIZ1
        NOP
        BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
BTIMEZ  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   BTIMEZ
        NOP
        NOP
        MOVLW  D'44'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  B'00011101'
        MOVWF  PORTB
BLOOPHZ DECFSZ DURHOR
        GOTO   BLOOPHZ
        MOVLW  B'00000001'
        MOVWF  PORTB
        NOP
        NOP
        NOP
        BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
```

```
BTIME2  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   BTIME2
        MOVLW  D'28'
        MOVWF  CANTHB1
        MOVLW  D'44'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  B'00011101'
        MOVWF  PORTB
BLOOPH5 DECFSZ DURHOR
        GOTO   BLOOPH5
        MOVLW  B'00000001'
        MOVWF  PORTB
        DECFSZ CANTHB2
        GOTO   BHORIZ1
        NOP

;ESTA ULTIMA MEDIA LINEA, LA 305, LA USO PARA CARGAR VARIABLES

        BCF    PORTB,0 ;BIT 0 PASA A NIVEL BAJO
        NOP ;PIERDO TIEMPO PARA
        NOP ;HACER LOS 4,8µS
        MOVLW  0 ;NO USAR EL BIT 1 DEL PORTB, BIT 0 = SYNC
        BTFSC  FIELD,0
        MOVLW  1
        MOVWF  CARRY
        NOP
        MOVLW  D'15'
        NOP
        NOP
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 PASA A NIVEL ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
BLOOPH6 DECFSZ DURHOR
        GOTO   BLOOPH6
        RRF    CARRY ;CARRY = 1 SI 1 H, CARRY = 0 SI 1/2 H
        BTFSC  PORTA,2
        GOTO   LECTURA
        BTFSS  PORTA,3
        GOTO   LECTURA
        NOP
        NOP
        NOP
        NOP
        NOP
        GOTO   INICIO2
```

```
;***** PUNTOS *****

INICIO3 RRF    FIELD
        NOP
        NOP
        MOVLW  D'4'
        MOVWF  BLKLIN
        MOVLW  D'28'
        MOVWF  CANTHB1
        MOVLW  D'10'
        MOVWF  CANTHB2
        MOVLW  4
        MOVWF  CANTPRE
        MOVLW  5
        MOVWF  CANTVER
        MOVLW  5
        MOVWF  CANTPOS
CPREEQU BCF    PORTB,0 ;DURACION: 2,6µS ABAJO
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
CLOOP1  DECFSZ DUREQU ;SE COMPLETAN LOS 32µS ARRIBA
        GOTO   CLOOP1
        NOP
        NOP
        DECFSZ CANTPRE
        GOTO   CPREEQU
        NOP
CVERT   BCF    PORTB,0
        MOVLW  D'22'
        MOVWF  DURVER
CLOOP2  DECFSZ DURVER
        GOTO   CLOOP2
        BSF    PORTB,0 ;DURACION: 4.8µS ARRIBA ("SERRATED PULSES")
        MOVLW  2
        MOVWF  TIEMPO
CTIME   DECFSZ TIEMPO
        GOTO   CTIME
        NOP
        DECFSZ CANTVER
        GOTO   CVERT
        NOP
CPOSEQU BCF    PORTB,0
        MOVLW  D'23'
        MOVWF  DUREQU
        NOP
        NOP
        NOP
        BSF    PORTB,0
CLOOP3  DECFSZ DUREQU
        GOTO   CLOOP3
        NOP
        NOP
        DECFSZ CANTPOS
        GOTO   CPOSEQU
```

```
        NOP
        NOP ;1/2 LINEA H
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVLW  D'21' ;TIEMPO PARA 1/2 H (80 CICLOS TOTAL)
        NOP
        NOP
        MOVWF  DURHOR
        NOP
        NOP
        NOP
        NOP
        BCF    PORTB,1
        NOP
CLOOP   DECFSZ DURHOR
        GOTO   CLOOP
        NOP
CHORIZ  BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
CTIME3  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   CTIME3
        NOP
        NOP
        MOVLW  D'48'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
CLOOPH3 DECFSZ DURHOR
        GOTO   CLOOPH3
        NOP
        DECFSZ BLKLIN
        GOTO   CHORIZ
        NOP
CHORIZ1 BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
CTIME1  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   CTIME1
        NOP
        NOP
        MOVLW  D'48'
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 ALTO
CLOOPHZ DECFSZ DURHOR
        GOTO   CLOOPHZ
        NOP
        DECFSZ CANTHB1
        GOTO   CHORIZ1
        NOP
        BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
```

```
CTIMEZ  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   CTIMEZ
        NOP
        NOP
        MOVLW  9
        MOVWF  CANTLIN
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
CLOOPHA MOVLW  B'00011100'
        ADDWF  PORTB
        SUBWF  PORTB
        NOP
        MOVLW  2
        MOVWF  DURHOR
CLOOPH4 DECFSZ DURHOR
        GOTO   CLOOPH4
        DECFSZ CANTLIN
        GOTO   CLOOPHA
        NOP
        MOVLW  B'00011100'
        ADDWF  PORTB
        SUBWF  PORTB
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        BCF    PORTB,0
        MOVLW  2
        MOVWF  TIEMPO ;PIERDO TIEMPO PARA
CTIME2  DECFSZ TIEMPO ;HACER LOS 4,8µS
        GOTO   CTIME2
        MOVLW  D'28'
        MOVWF  CANTHB1
        MOVLW  9
        MOVWF  CANTLIN
        BSF    PORTB,0 ;BIT 0 ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
```

```
CLOOPHB MOVLW  B'00011100'
        ADDWF  PORTB
        SUBWF  PORTB
        NOP
        MOVLW  2
        MOVWF  DURHOR
CLOOPH5 DECFSZ DURHOR
        GOTO   CLOOPH5
        DECFSZ CANTLIN
        GOTO   CLOOPHB
        NOP
        MOVLW  B'00011100'
        ADDWF  PORTB
        SUBWF  PORTB
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        DECFSZ CANTHB2
        GOTO   CHORIZ1
        NOP

;ESTA ULTIMA MEDIA LINEA, LA 305, LA USO PARA CARGAR VARIABLES

        BCF    PORTB,0 ;BIT 0 PASA A NIVEL BAJO
        NOP ;PIERDO TIEMPO PARA
        NOP ;HACER LOS 4,8µS
        MOVLW  0 ;NO USAR EL BIT 1 DEL PORTB, BIT 0 = SYNC
        BTFSC  FIELD,0
        MOVLW  1
        MOVWF  CARRY
        NOP
        MOVLW  D'15'
        NOP
        NOP
        MOVWF  DURHOR
        BSF    PORTB,0 ;BIT 0 PASA A NIVEL ALTO
        NOP
        NOP
        NOP
        NOP
        NOP
CLOOPH6 DECFSZ DURHOR
        GOTO   CLOOPH6
        RRF    CARRY ;CARRY = 1 SI 1 H, CARRY = 0 SI 1/2 H
        BTFSS  PORTA,2
        GOTO   LECTURA
        BTFSS  PORTA,3
        GOTO   LECTURA
        NOP
        NOP
        NOP
        NOP
        NOP
        GOTO   INICIO3

        END
```

CAPITULO 4:
MEMORIA DESCRIPTIVA
Y DE CALCULO.

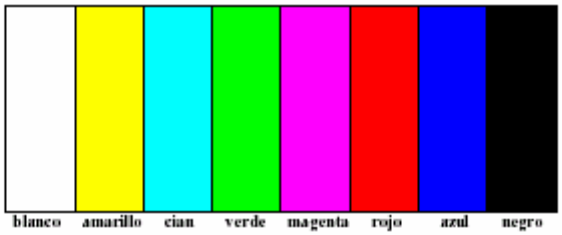
4.1. DESCRIPCIÓN Y FUNCIONAMIENTO DEL GENERADOR.

DESCRIPCIÓN Y FUNCIONAMIENTO DEL GENERADOR

Según lo visto en el apartado de colorimetría, para realizar un sistema de televisión en color bastaría con transmitir las componentes de color. Entonces un generador de barras básico podría tener tres salidas, una para cada color primario, y cada una de estas salidas se conectaría a la correspondiente entrada del televisor.

El equipo generaría combinaciones de sus salidas, según la siguiente tabla:

Azul	1	0	1	0	1	0	1	0
Rojo	1	1	0	0	1	1	0	0
Verde	1	1	1	1	0	0	0	0



En esta tabla un “1” significa “presencia” del color, en tanto que un “0” es su “ausencia”. Como se observa, es muy sencillo construir un generador de este tipo, ya que bastaría un mínimo de electrónica digital para obtener estas barras, pero la mayoría de los equipos de televisión y video no poseen entradas directas de Rojo, Verde y Azul, quedando estas reservadas para monitores profesionales.

Lo habitual es que los equipos domésticos tengan una entrada de "Vídeo Compuesto", denominada usualmente como "VÍDEO IN". Por esta razón, nuestro generador debe convertir las componentes Roja, Verde y Azul en la señal de "Vídeo Compuesto".

Un generador de patrones no solo debe entregar las componentes RGB de la señal, sino que debe generar video compuesto, con todo lo que esto implica: generar sincronismos, obtener la señal de luminancia, generar una subportadora de color y modularla, etc.

Lo primero que debe definirse es la cantidad y tipo de patrones que deberá generar el equipo, los controles que habrá de poseer, y las especificaciones del mismo, ya que esto determinará las características, y por tanto la complejidad, del sistema a desarrollar.

Especificaciones del sistema.

El equipo será capaz de generar siete patrones básicos:

1) Barras de color.
2) Barrido en blanco.
3) Barrido en Rojo.
4) Barrido en Verde.
5) Barrido en Azul.
6) Damero.
7) Rejilla.

Se requieren controles independientes para las señales R, G, B, luminancia, y crominancia. De este modo se amplía la cantidad de patrones que pueden ser generados, ya que el Barrido se podrá hacer con cualquiera de los 8 colores de la carta de barras, y las Barras podrán ser monocromáticas o adoptar diferentes combinaciones de color. Se permitirá la supresión del BURST de color, herramienta útil en la detección de fallos relacionadas con los circuitos de proceso de color. Por ultimo también se requiere un control para el nivel de salida. Todo esto se puede ver en el diagrama de bloques de la figura 3.1).

Será necesario una salida de Vídeo Compuesto que deberá entregar 1Vpp en una carga de 75Ω, siendo opcional una salida de Súper Vídeo (Luminancia y Crominancia).

El barrido será entrelazado en todos los modos de funcionamiento y según las normas de exploración de nuestro país (Normas B y G, recogidas en la Rec. UIT-R BT 470-2).

Normas de Exploración.

Una norma es un conjunto de parámetros adoptados como regla dentro de determinado grupo o región a fin de mantener una relación clara y sin ambigüedades entre las partes. En televisión es exactamente eso. Las normas establecen los parámetros que deben seguir tanto los equipos transmisores de señal como los receptores, a fin de que se establezca una comunicación segura y sin errores entre ambos. Entre los muchos parámetros que se fijan consideraremos solo los relacionados con nuestro proyecto.

Normas B y G.

Actualmente en el mundo hay muchas normas de transmisión de televisión, denominadas con letras desde la "A" a la "N". Algunas ya no se utilizan, pero la mayoría siguen vigentes.

Veamos ahora los valores que se han establecido para los parámetros en la norma de nuestro país:

Nº de líneas/cuadro:	625 líneas.
Nº de cuadros/sg:	fp = 25 cuadros/sg.
Nº de campos/sg:	fc = 50 campos/sg.
Relación entrelazado:	2/1.
Nº de líneas/sg:	fh = 15.625.
Relación de aspecto:	A = 4/3.
Nº de líneas activas:	575.
Factor Resol. horizontal:	80 líneas/MHz.
Dirección de exploración:	Horizontal: De izquierda a derecha.
	Vertical: De arriba a abajo.

Solución para el diseño del generador.

Para cumplir las especificaciones se decidió incluir los siguientes bloques:

1) **Microcontrolador PIC16F84-10**.

Procesa la información que recibe del teclado, y genera las componentes de color y la señal de sincronismo compuesto.

Para cumplir estas tareas hace lo siguiente:

- Acepta en uno de sus terminales, configurado como entrada, las órdenes provenientes de un pulsador, para de este modo poder seleccionar el patrón a generar. Esta entrada se corresponde con el Bit 2 del PORTA.
- Genera una base de tiempos estable, de donde obtener todos los tiempos requeridos por los sincronismos.
- Genera en uno de sus terminales, el correspondiente al Bit 0 del PORTB, todos los sincronismos requeridos por la norma de televisión adoptada, sin agregar video a esta señal (sincronismo puro).
- Genera en tres terminales las señales R, G y B, que correspondan con el patrón que deba mostrarse a la salida. Estos terminales no tendrán sincronismos (video puro). La designación de terminales es la siguiente: PORTB (2) = B, PORTB (3) = R , y PORTB (4) = G.

2) **Conversor CXA1645P.**

Combina las señales RGB con la señal de sincronismo compuesto, y genera la señal de vídeo compuesto.

La conversión se realiza de la siguiente manera:

- Posee cinco entradas de señal: Sincronismo, R, G, B, y Subportadora de Color (SC).
- A partir de RGB genera la señal de luminancia (Y).
- Genera las señales B-Y y R-Y, con la alternancia de fase requerida por el sistema PAL.
- A partir de B-Y, R-Y, y SC genera la señal de crominancia (C).
- Mezcla Y con C para obtener Vídeo Compuesto.

Con este circuito integrado de SONY se obtienen salidas de vídeo compuesto y de súper vídeo.

La circuiteria externa que se requiere es mínima, ya que se integran casi la totalidad de los circuitos, dando esto lugar a una gran sencillez en el diseño y una gran calidad en las señales. Pero de esta forma solo se cumplen parte de las especificaciones, ya que la integración de circuitos impide el acceso a gran parte de las señales, y de esta forma la posibilidad de controlarlas. La solución adoptada consiste en emplear otro conversor de vídeo, conectado en cascada con el anterior, en el que se puedan manipular las señales. Este nuevo conversor de vídeo se describe a continuación.

3) **Conversor MC1377P.**

De la misma forma que en el caso anterior combina las señales RGB con la señal de sincronismo compuesto, y genera la señal de vídeo compuesto.

La conversión se realiza de la misma forma, salvo que en este caso el oscilador de crominancia también se encuentra integrado.

Con este circuito integrado de MOTOROLA se obtiene solo una salida de vídeo compuesto, y ha diferencia del caso anterior se requiere cierta circuiteria externa. Esta circuiteria externa consiste en un filtro para la señal de crominancia, un red de retardo para la señal de luminancia, y un circuito de control para el disparo de la salva de color. Como puede verse se tiene acceso a todas las señales importantes, y esa cualidad es la que vamos ha aprovechar, para introducir circuitos que las controlen, y de esa forma cumplir las especificaciones que faltaban.

A continuación, se comentaran cada uno de los bloques, describiendo detalladamente las características de cada uno y las conexiones de estos con el resto del circuito.

El software se comentará más adelante en el capitulo 5.

4.1.1. MICROCONTROLADOR PIC16F84.

4.1.1.1. Características técnicas.

Se trata de un microcontrolador de 8 bits, con memoria de tipo Flash/EEPROM, y cápsula de 18 terminales.

Las principales características proporcionadas por el fabricante son las siguientes:

Características de la CPU:

- CPU tipo RISC de altas prestaciones.
- Solo 35 instrucciones.
- Todas las instrucciones se ejecutan en un solo ciclo, excepto las instrucciones de salto, que requieren dos ciclos.
- Velocidad de operación:
  - Entrada de reloj -> DC – 10MHz.
  - Ciclo de instrucción -> DC – 400ns.
- 1K de memoria de programa tipo Flash.
- 68 Bytes de memoria de datos tipo RAM .
- 64 Bytes de memoria de datos tipo EEPROM.
- Anchura de Instrucciones de 14 bits.
- Anchura de Datos de 8 bits.
- 15 registros hardware para operaciones especiales.
- Pila hardware de 8 niveles.
- Modos de direccionamiento directo, indirecto, y relativo.
- 4 fuentes de interrupción:
  - Externa a través de patilla RB0/INT.
  - Por desbordamiento del temporizador TMR0.
  - Por cambio de estado en los bits 4-7 de PORTB.
  - Por fin de escritura en la memoria de datos EEPROM.
- Memoria de programa Flash con 1000 ciclos de lectura/escritura.
- Memoria de datos EEPROM con 10.000.000 de ciclos de lectura/escritura.
- Retención de datos en la memoria EEPROM durante más de 40 años.

Características de los Periféricos:

- 13 terminales de entrada/salida con control individual de dirección.
- Alta corriente de entrada/salida con capacidad de manejo de diodos LED.
  - 25 mA de entrada máximo por terminal.
  - 20 mA de salida máximo por terminal.
- Temporizador/Contador de 8 bits con divisor de 8 bits programable.

Características especiales del Microcontrolador:

- Programación serie en circuito a través de dos terminales (ICSP).
- Power-on Reset (POR).
- Power-up Timer (PWRT).
- Oscillator Start-up Timer (OST).
- Perro guardián.
- Protección del código.
- Modo reposo de bajo consumo.
- Oscilador seleccionable.

Características eléctricas:

- Temperatura ambiente bajo polarización.....-55°C a +125°C
- Temperatura de almacenaje.....-65°C a +150°C
- Voltaje en VDD con respecto a VSS.....-0.3 a +7.5V
- Voltaje en MCLR con respecto a VSS..... -0.3 to +14V
- Voltaje en algún pin con respecto a VSS (excepto VDD y MCLR)..... -0.6V a (VDD + 0.6V)
- Disipación de potencia total.....800 mW
- Máxima corriente de salida por VSS.....150 mA
- Máxima corriente de entrada por VDD.....100 mA
- Corriente de entrada por borne, IK (VI < 0 or VI > VDD).....± 20 mA
- Corriente de salida por borne, IOK (VO < 0 or VO > VDD) .....± 20 mA

- Máxima corriente de entrada por algún terminal de I/O..... 25 mA
- Máxima corriente de salida por algún terminal de I/O.....20 mA
- Máxima corriente de entrada por PORTA.....80 mA
- Máxima corriente de salida por PORTA.....50 mA
- Máxima corriente de entrada por PORTB.....150 mA
- Máxima corriente de salida por PORTB.....100 mA

Tabla 4.1 - Descripción de terminales del microcontrolador PIC16F84.

Nombre	Número	Tipo I/O/P	Tipo Buffer	Descripción
OSC1/CLKIN	16	I	ST/CMOS	Entrada oscilador cristal/ entrada reloj externo.
OSC2/CLKOUT	15	O	-	Salida oscilador cristal/ salida ¼ reloj externo.
/MCLR	4	I/P	ST	Reset principal.
RA0	17	I/O	TTL	Puerto de entrada/salida bidireccional.
RA1	18	I/O	TTL	
RA2	1	I/O	TTL	
RA3	2	I/O	TTL	
RA4/T0CKI	3	I/O	ST	También entrada de reloj del TMR0.
RB0/INT	6	I/O	TTL/ST	Puerto de entrada/salida bidireccional. Resistencias de pull-up. RB0 capacidad de INT. RB4-7 capacidad de INT. por cambio de estado. RB6-RB7 reloj-datos programación serie
RB1	7	I/O	TTL	
RB2	8	I/O	TTL	
RB3	9	I/O	TTL	
RB4	10	I/O	TTL	
RB5	11	I/O	TTL	
RB6	12	I/O	TTL/ST	
RB7	13	I/O	TTL/ST	
VSS	5	P	-	Tierra
VDD	14	P	-	Alimentación positiva

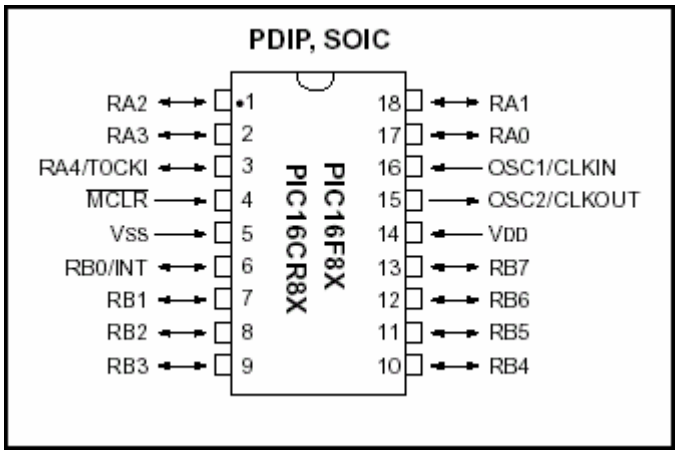


Figura 4.1 – Diagrama de terminales del microcontrolador PIC16F84.

4.1.1.2. Arquitectura general de la gama baja.

Introducción.

La configuración básica de los microcontroladores PIC de la gama baja es similar a la de las restantes gamas. Por tal motivo, se comienza describiendo la estructura y funcionamiento de la misma, dejando para un apartado posterior las novedades y mejoras que incorporan los elementos más complejos.

Uno de los pilares en los que se basa la organización de los PIC es la arquitectura Harvard. Con ella, la UCP accede de forma independiente y simultánea a la memoria de datos y a la de instrucciones. Este aislamiento entre datos e instrucciones permite que cada uno tenga el tamaño más adecuado. Así los datos tienen una longitud 8 de bits, mientras que las instrucciones la tienen de 12 bits.

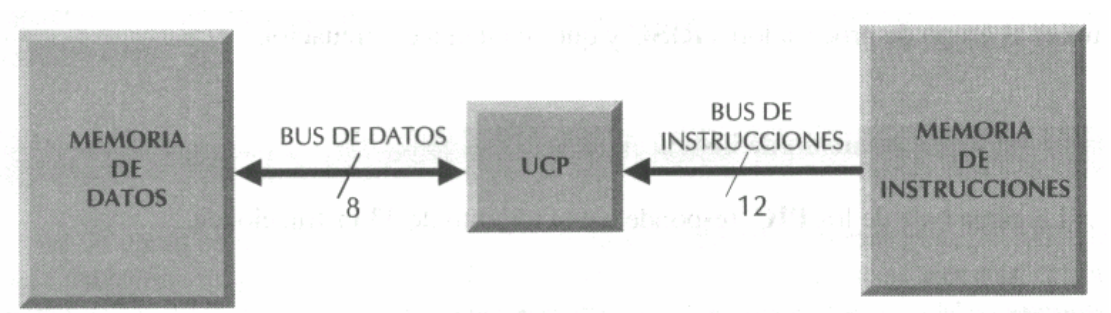


Figura 4.2 – Anchura de los buses de datos e instrucciones.

Otro de los recursos que propicia el manejo intensivo de la arquitectura Hardvard es el "Banco de Registros", que participa de manera muy flexible en la ejecución de las instrucciones. Como se muestra en la figura 4.3, la ALU realiza sus operaciones lógico-aritméticas con dos operandos, uno que recibe desde el registro W, que a hace las veces de Acumulador de los microprocesadores convencionales, y otro que puede provenir de cualquier otro registro interno. El resultado de la operación se puede depositar en cualquier registro.

Esta funcionalidad da a conocer un carácter completamente "ortogonal" a las instrucciones, posibilitando que los operandos fuente y destino estén ubicados en casi cualquier registro.

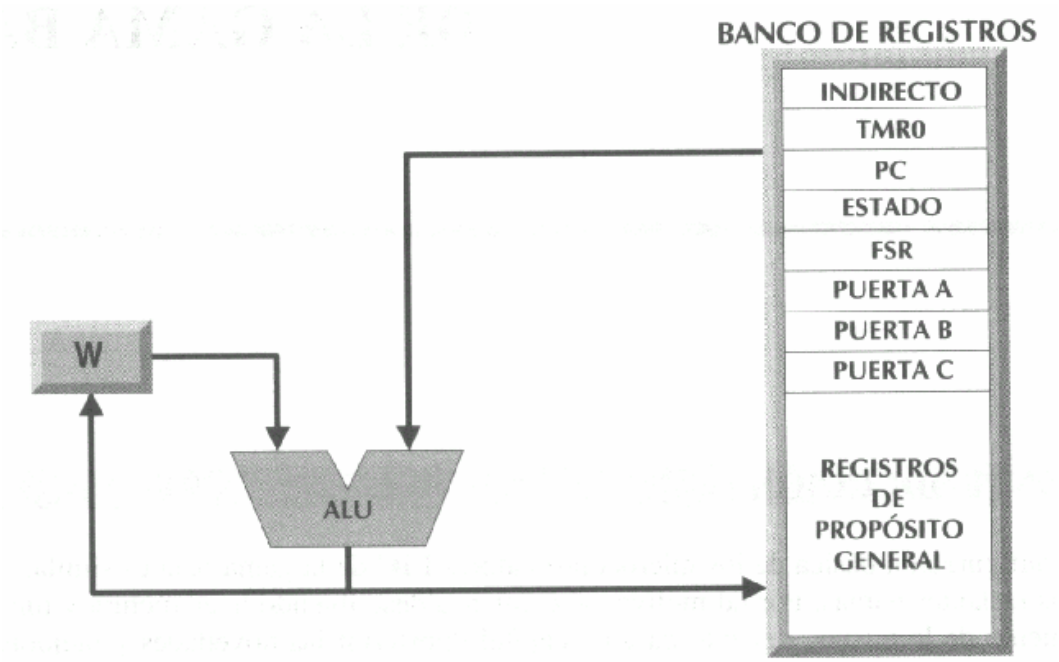


Figura 4.3 – La ALU recibe los operandos desde el registro W y de cualquier otro del banco general de registros. El resultado puede depositarse en cualquier registro.

Los microcontroladores PIC reúnen todas las condiciones necesarias para pertenecer al grupo de procesadores RISC y que se citan a continuación:

- Juego reducido de instrucciones: La gama baja de los PIC responde a un conjunto de 33 instrucciones.
- Idéntico formato de las instrucciones: Toda las instrucciones de la gama baja tienen una longitud de código de 12 bits.
- Ortogonalidad: Las instrucciones pueden utilizar cualquier objeto como operando fuente o destino.
- Ejecución de instrucciones simples en un ciclo: Todas las instrucciones de los PIC se ejecutan en un ciclo, menos las de salto que tardan dos.

Las instrucciones disponen de tres modos de direccionado: directo, indirecto y relativo. La pila que guarda los retornos al programa principal solo dispone de dos niveles en los componentes de la gama baja, lo cual constituye una seria limitación en la confección del software ya que no se pueden anidar más de dos subrutinas consecutivamente.

La tecnología CMOS con la que se fabrican los PIC permite que la tensión de alimentación pueda oscilar entre 2 y 6,25 V con un reducido consumo. A 5 V y 4 MHz, el consumo típico es menor que 2 mA y a 3 V y 32 kHz, es inferior a 15 pA. Cuando el microcontrolador funciona en "modo de reposo o espera" y con el temporizador Perro guardián desconectado la potencia necesaria es menor de 3 pA. Todas estas características posibilitan en muchas ocasiones la alimentación de los sistemas con pilas convencionales.

El reloj y el ciclo de instrucción.

En la gama baja, la frecuencia máxima de la señal de reloj interna es de 20 MHz lo que determina un periodo de 50 ns. El ciclo de instrucción en el que se ejecutan la mayoría de las instrucciones se compone de cuatro ciclos de reloj, que a 20 MHz suponen una duración de 200 ns.

En realidad, cada instrucción conlleva dos ciclos de instrucción, el primero destinado a la "fase de búsqueda" y el otro, a la "fase de ejecución". Sin embargo, la estructura segmentada del procesador permite realizar simultáneamente la fase de ejecución de una instrucción y la de búsqueda de la siguiente.

Cuando la instrucción ejecutada corresponde a un salto, no se conoce cuál será la siguiente hasta completarla por lo cual en esta situación se sustituye la fase de búsqueda por una instrucción NOP (No Operar) mientras se ejecuta un salto. Esta característica se muestra gráficamente en la figura 4.4 y en ella se puede apreciar cómo las instrucciones de salto precisan un ciclo más.

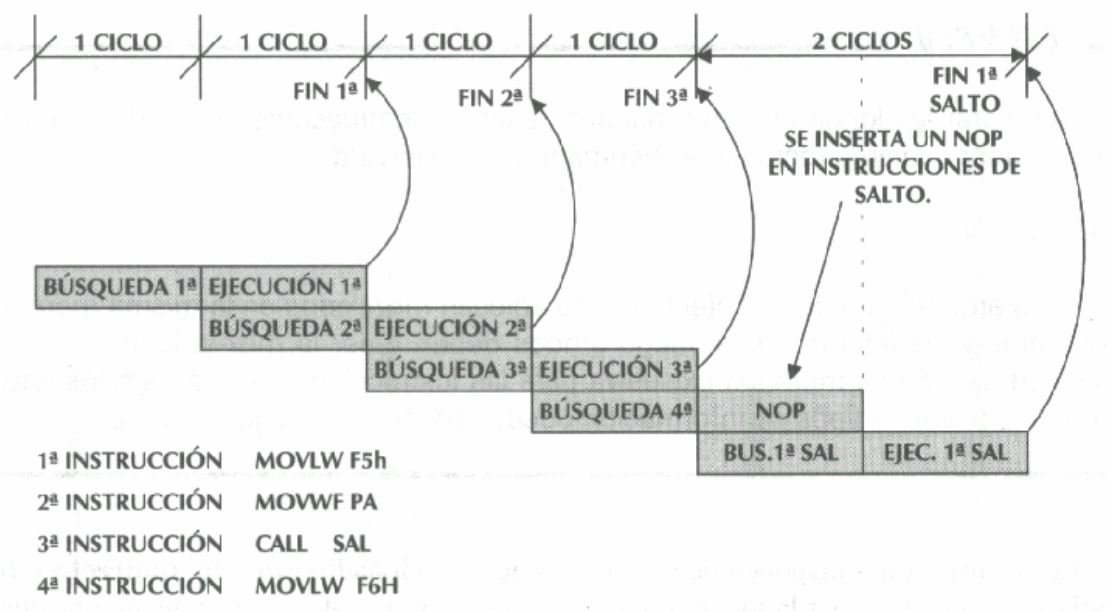


Figura 4.4 – La segmentación permite completar las instrucciones en un solo ciclo excepto las de salto que necesitan dos.

Para el funcionamiento del circuito de reloj interno se precisa colocar en el exterior una fuente de señal con una frecuencia fija disponiendo los microcontroladores PICl6C5X de dos patitas para soportar dicha señal :

OSC1/CLKIN: Es la patita a la que se conecta la señal de entrada de la fuente externa de frecuencia, que puede estar implementada por un cristal de cuarzo, por un resonador cerámico o una red RC.

OSC2/CLKOUT: Se trata de la patita de conexión de la salida del cristal externo. En el modo oscilador RC por esta patita sale la cuarta parte de la frecuencia de oscilación, delimitando los ciclos de instrucción. Para esta alternativa Microchip recomienda usar una Rext con un valor comprendido entre 5 kΩ y 100 kΩ y un Cext con más de 20pF. Figura 4.5.

Los PIC pueden funcionar con cuatro tipos de osciladores. En las versiones EPROM, con ventana, y en las OTP, QTP y ROM sin ella, hay que distinguir el chip específico para el oscilador, que viene marcado en el encapsulado. Esto significa que los PIC se comercializan con cuatro referencias distintas, que corresponden con los tipos de reloj. Se debe grabar el utilizado escribiendo el código adecuado en los dos bits FOSC1 y FOSC0 de la "Palabra de Configuración" que se comenta más adelante. Dichos códigos son los siguientes :

- 11 -> RC. Oscilador RC de bajo coste. Figura 4.5. Este oscilador proporciona una estabilidad de frecuencia mediana.
- 10 -> HS. Oscilador de alta velocidad (8-20MHz).
- 01 -> XT. Oscilador estándar (100KHz-4MHz). Cuarzo estándar o resonador cerámico.
- 00 -> LP. Oscilador de bajo consumo (32-200KHz). Cristal de cuarzo o resonador para aplicaciones de muy bajo consumo.

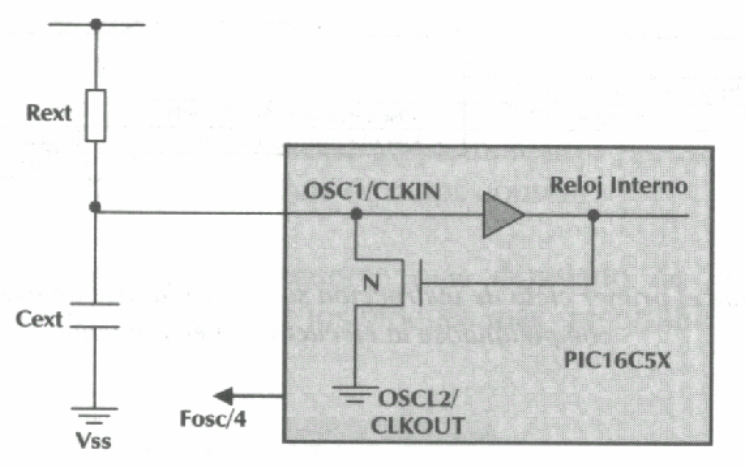
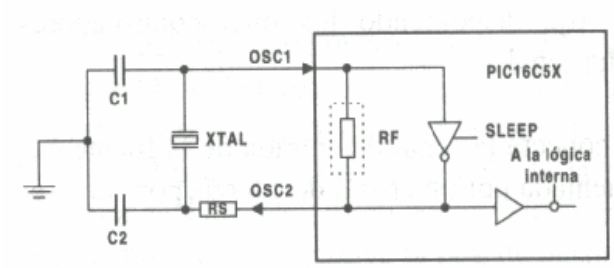


Figura 4.5 – Esquema de conexionado de un oscilador de bajo coste.



Tipo de oscilador	Frecuencia del cristal	Rango de C1	Rango de C2
LP	33 kHz 200 kHz	33-68 pF 15-33 pF	33-68 pF 15-33 pF
XT	100 kHz 2 MHz 4 MHz	68-100 pF 10-22 pF 10-22 pF	68-100 pF 10-22 pF 10-22 pF
HS	8 MHz 20 MHz	22-47 pF 22-47 pF	22-47 pF 22-47 pF

Figura 4.6 – Según el tipo de oscilador usado y la frecuencia de trabajo, se emplean diferentes valores en los condensadores C1 y C2 que acompañan al cristal de cuarzo. La resistencia RS solo es necesaria en algunas versiones HS.

La frecuencia de la señal de entrada aplicada a la patita OSC1/CLKIN se divide internamente conformando cuatro estados que delimitan cada ciclo de instrucción y que se denominan Ql-Q4. Durante Ql, el valor del PC (Contador de Programa) se incrementa. En el transcurso de Q4 se efectúa la búsqueda de la instrucción en la memoria y su código se carga en el Registro de Instrucciones. En el siguiente ciclo de instrucción se llevan a cabo la decodificación y ejecución de la instrucción. Figura 4.7.

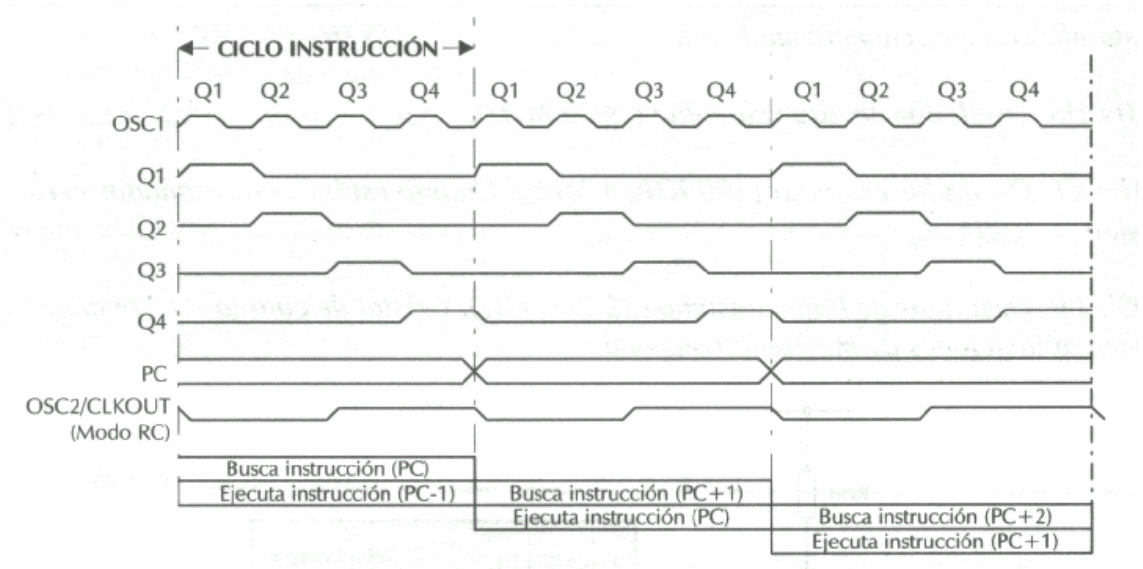


Figura 4.7 – Durante el primer ciclo de instrucción se realiza la fase de búsquela de la instrucción, completándose la ejecución en el siguiente.

Descripción de la arquitectura básica.

En la figura 4.8 se ofrece el esquema general muy simplificado de la estructura interna de los microcontroladores PIC de la gama baja.

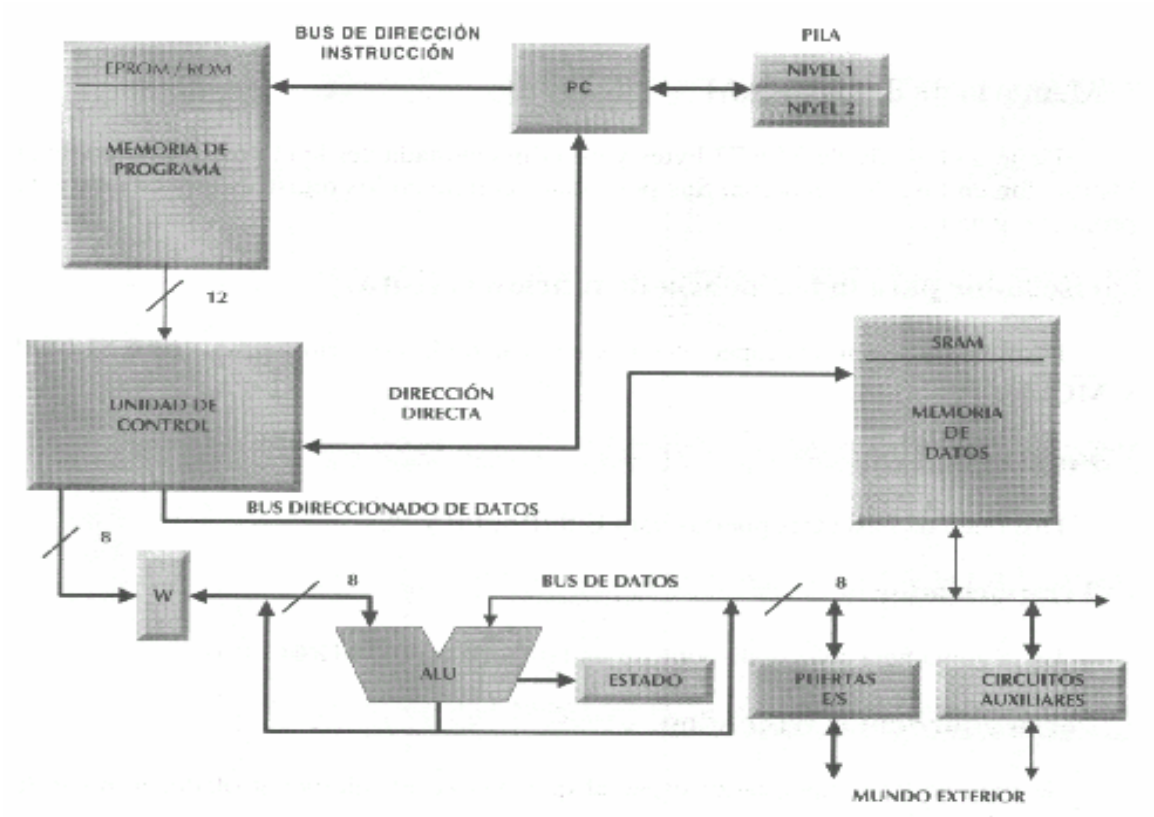


Figura 4.8 – Esquema simplificado de la arquitectura básica de los PIC de la gama baja.

El microcontrolador funciona bajo el control de las instrucciones grabadas en la memoria de instrucciones, no volátil. El PC proporciona la dirección de la instrucción en curso y su código de 12 bits se introduce en la Unidad de Control donde se decodifica y se generan las órdenes para la ejecución de la misma. Según el tipo de instrucción se localizan los operadores que se aplican a las entradas de la ALU. Uno de ellos puede ser el contenido del registro W y el otro puede llegar por el bus de datos desde cualquier elemento. Todos los objetos (puertas de E/S, temporizadores, posiciones de memoria, etc) se manejan como si estuviesen implementados físicamente por registros, y su lectura y escritura se efectúa por el bus de datos común.

Reset.

El reset puede ser originado por las siguientes causas:

- Conexión de la alimentación (POR: Power-On Reset).
- Activación de la patita /MCLR (Master Clear Reset) durante una operación normal.
- Activación de /MCLR en el estado de Reposo o SEP.
- Desbordamiento del Perro guardián.

Como se aprecia en el esquema de la figura 4.9, cualquiera de estas posibilidades introduce un nivel bajo en la entrada S del flip-flop y pone en marcha un temporizador propio que, al cabo de 18 ms, origina un flanco ascendente en la salida /Q que supone la generación del reset interno.

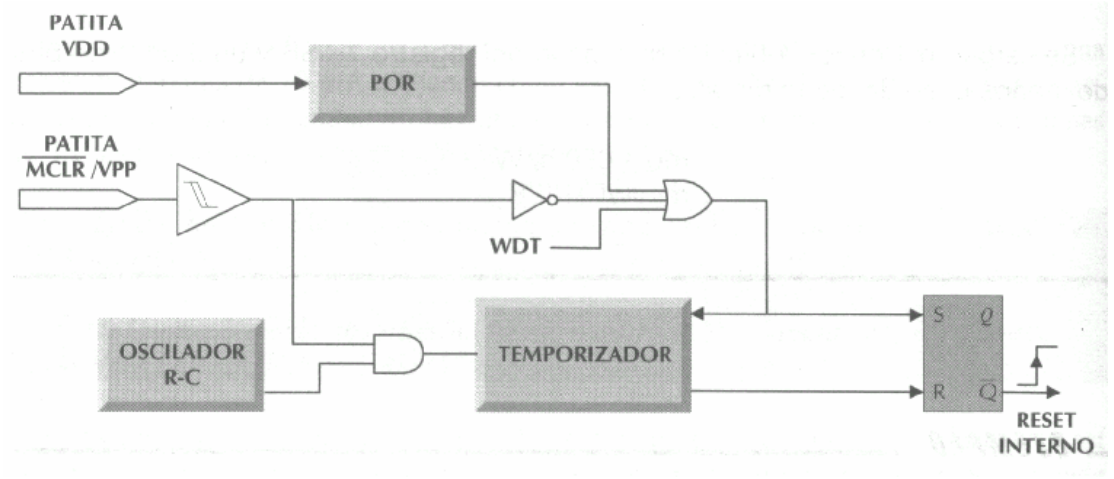


Figura 4.9 – Esquema por bloques del circuito que controla la activación de reset interno.

El bloque temporizador de la figura 4.9 produce un retraso de 18 ms en la generación del reset para dar tiempo a que se estabilice la tensión VDD de alimentación y la frecuencia del oscilador principal. Este temporizador está gobernado por un oscilador RC independiente.

Los bits /TO y /PD del Registro de Estado toman el valor correspondiente según la causa que haya provocado el reset.

4.1.1.3. Organización de la memoria en la gama baja.

La repercusión más importante del empleo de la arquitectura Harvard en los microcontroladores PIC se manifiesta en la organización de la memoria del sistema. La memoria de programa es independiente de la de los datos, teniendo tamaños y longitudes de palabra diferentes.

En los PICl6C5X el formato de todas las instrucciones es de 12 bits y, en consecuencia, la longitud de las palabras de la memoria de programa también. Este tamaño permite codificar en una palabra el código OP de la instrucción junto al operando o su dirección. Para adaptarse a las necesidades de las aplicaciones del usuario hay modelos de la gama baja con 512 posiciones para la memoria de instrucciones y otros que tienen 1 k y 2 k posiciones de 12 bits.

Puesto que los datos y operandos que manejan las instrucciones son de 8 bits, la longitud de las palabras de la memoria de datos tiene ese tamaño. La capacidad de la SRAM varía entre 25 y 73 posiciones, según el modelo.

En la figura 4.10 se ofrece el esquema general de adaptación de la UCP con las memorias de los PICl6C5X. El bus que direcciona las posiciones de la memoria de programa tiene un tamaño de 9 a 11 líneas para soportar capacidades comprendidas entre 512 y 2.048. El bus que direcciona la memoria de datos dispone de cinco a siete líneas para seleccionar entre un mínimo de 25 bytes y un máximo de 73.

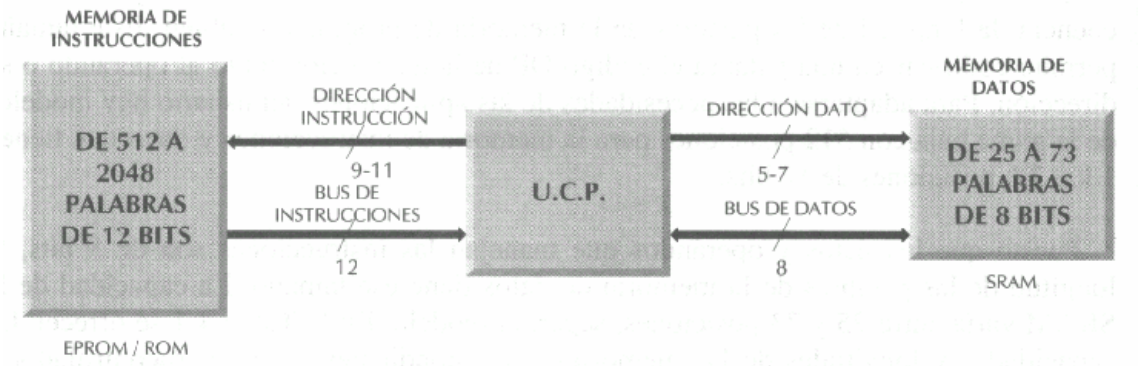


Figura 4.10 – Esquema general de conexionado de la UCP con las memorias de instrucciones y datos.

El tamaño de los buses que direccionan la memoria de datos y la de programa son diferentes. Lo mismo pasa con el bus que transfiere las instrucciones y el que lo hace con los datos. La total independencia entre los accesos a las dos memorias permite realizar accesos simultáneos.

La memoria de programa siempre está direccionada desde el Contador de Programa (PC), mientras que la memoria de datos puede direccionarse directamente desde parte de1 código OP de la instrucción o indirectamente a través de un registro denominado FSR (Registro de Selección del Banco). Figura 4.11.

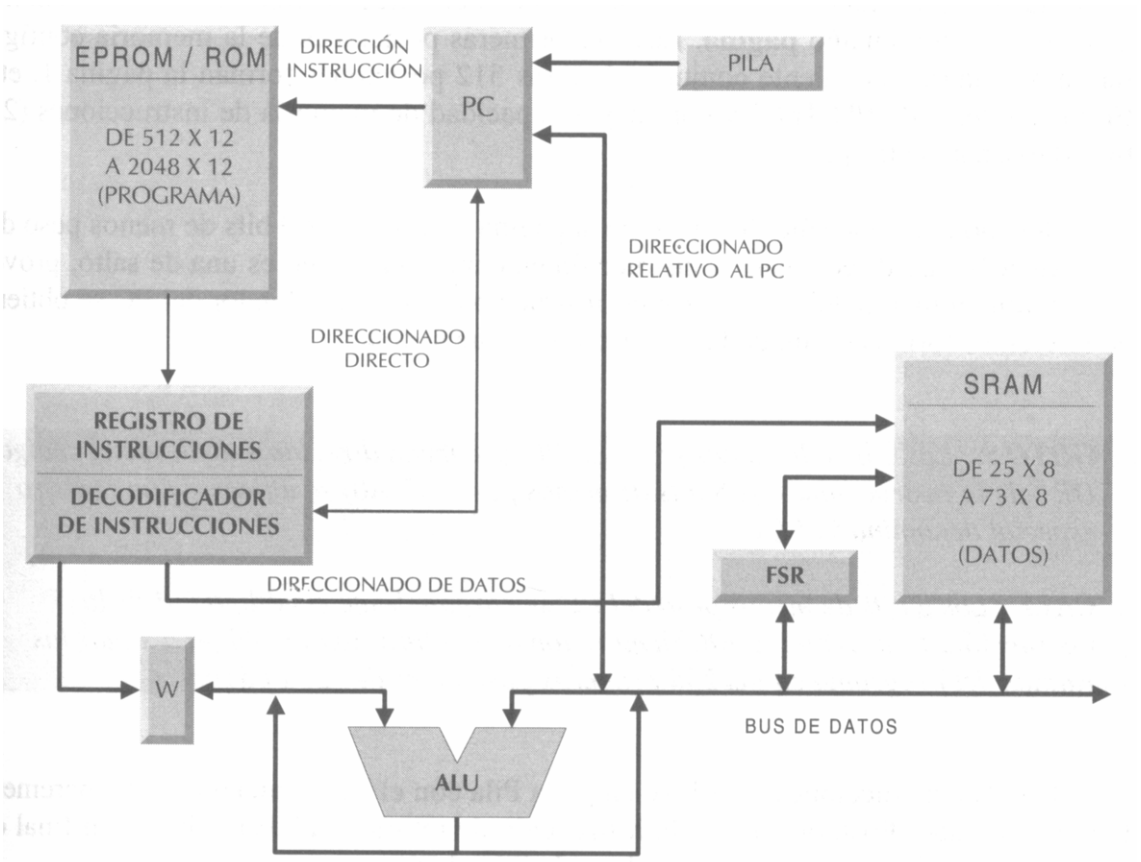


Figura 4.11 – Direccionamiento de las memorias de programa y datos en los PIC de la gama baja.

Memoria de Programa.

La memoria de instrucciones puede tener una capacidad mínima de 512 palabras de 12 bits hasta una máxima de 2.048 palabras de la misma longitud. Durante la fase de búsqueda, la dirección de la instrucción la proporciona el PC, el cual normalmente se autoincrementa en la mayoría de las instrucciones, excepto en las de salto.

Hasta 512 posiciones la memoria se direccionan directamente con 9 bits, denominándose a dicho tamaño página. Las 512 primeras posiciones de la memoria configuran la página 0, el siguiente conjunto de otras 512 posiciones forman la página 1, etc. En los modelos de PICl6C5X con mayor capacidad de memoria de instrucciones (2 kbytes) existen cuatro páginas.

En los modelos que sólo disponen de la página 0, bastan los 9 bits de menos peso del PC (A8-AO) para direccionar la instrucción en curso, que, si no es una de salto, provoca el autoincremento del mismo. En las instrucciones de salto el valor del PC se obtiene de diferentes formas según de la instrucción de que se trate:

- GOTO: Los 9 bits de menos peso del PC se cargan directamente desde el código OP de la instrucción. Los 8 bits de menos peso (A7-A0) se ubican en un registro especial denominado PCL.
- CALL: Los 8 bits de menos peso del PC se cargan desde el código OP de la instrucción. El noveno bit, A8, siempre toma el valor 0 lo que obliga a situar las rutinas del programa en las 256 primeras posiciones de cada página.

Con las instrucciones CALL se carga la Pila con el valor inicial del PC incrementado una unidad. Posteriormente, la instrucción de retorno RETWL situada al final de la rutina, devuelve el valor almacenado en la Pila al PC.

El Contador de Programa.

El PC proporciona la dirección de la memoria de programa y su longitud puede oscilar entre 9 y 11 bits, según la capacidad de la misma. En la figura 4.12 se muestra la longitud que adopta el PC para los diversos modelos de PICl6C5X.

Los 8 bits de menos peso del PC (A07-A00) están implementados físicamente en un registro denominado PCL, que se ubica en la dirección O2 de la memoria de datos.

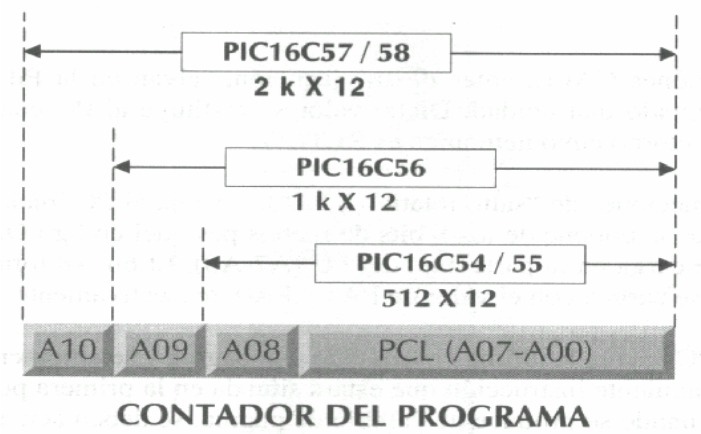


Figura 4.12 – Longitud que adopta el PC para los diversos modelos de PIC de la gama baja.

El comportamiento del PC para las diversas instrucciones del repertorio es el siguiente:

1°. Para todas las instrucciones que no entrañen saltos en el programa, el PC se autoincrementa para apuntar la siguiente instrucción a la que se está ejecutando.

2°. Las instrucciones de salto GOTO cargan en el PC directamente los 9 bits de menos peso desde el código OP de la propia instrucción. Cuando el área de programa es superior a las 512 posiciones, los 2 bits de más peso del PC (A10 y A9) se cargan con el valor de los bits PA1 y PAO del Registro de Estado.

3°. Las instrucciones CALL contienen en su código OP el valor de los 8 bits de menos peso del PC. El bit A8 toma en este caso el valor 0 lo que obliga a restringir la ubicación de las rutinas a las 256 primeras posiciones de cada página. Cuando son necesarios los bits A10 y A9, se cargan con los bits PA1 y PAO del Registro de Estado.

Las instrucciones CALL, antes de su ejecución, salvan en la Pila el valor del PC inicial incrementado una unidad. Dicho valor se restituye al PC cuando se ejecuta la instrucción de retorno cuyo nemónico es RETLW.

4°. En las instrucciones de "salto relativo", la ALU suma al PC inicial (A7-AO), el valor del salto que se obtiene de los 8 bits de menos peso del código OP. El resultado de la operación se carga en la parte baja del PC (A7-AO). El bit A8 toma el valor 0 y los bits A10 y A9 se cargan con el valor de PA1 y PAO, respectivamente.

Cuando el PC apunta la última posición de una página y se autoincrementa pasa a direccionar a la siguiente instrucción que estará situada en la primera posición de la página siguiente. Cuando se produce un cambio de página, se deben actualizar los bits PA1 y PAO del Registro de Estado porque, en caso contrario, cuando se ejecute una instrucción de salto, éste seguirá controlando la página en la que se halla esta instrucción de salto.

Cuando se genera un reset, los bits PA1 y PAO del Registro de Estado toman el valor 0. El PC queda apuntando la última posición de la última página que exista en la EPROM. Si en dicha posición se coloca una instrucción de GOTO, se pasa a direccionar una posición de la página 0.

En el caso de llevarse a cabo un reset en un microcontrolador con 512 posiciones de programa, la primera instrucción que se ejecuta es la que ocupa la dirección 1FF h en el PICl6C54/55. Si tiene 1 k posiciones , como sucede al PICl6C56, la primera instrucción que se ejecuta tras un reset es la que ocupa la dirección 3FF h y, si tiene 2 k (PICl6C57/58), ocupa la dirección 7FF h.

Memoria de Datos.

Los microcontroladores PIC funcionan con datos de 8 bits por lo que las posiciones de la memoria de datos tienen esa longitud. La capacidad de esta memoria varía entre 25 y 73 posiciones de un byte, en los modelos de la gama baja.

La memoria de datos se organiza en "bancos", pudiendo existir hasta cuatro en los modelos de mayor capacidad. El banco 0 tiene un tamaño de 32 bytes, mientras que todos los demás sólo disponen de 16. El banco 0 ocupa las 32 primeras posiciones de la memoria y lo poseen todos los modelos de PICl6C5X. Las 16 posiciones de menos peso de los bancos 1, 2 y 3 no son accesibles y cuando se direcciona desde el valor 00 h al OF h, siempre se acude al banco 0. Cuando la dirección está comprendida entre la 10 h y la lF h, hay que seleccionar a cuál de los cuatro posibles bancos corresponde con los bits 6 y 5 del registro FSR.

La memoria de datos funciona de forma similar al "banco de registros" de un procesador por lo cual sus posiciones implementan registros de propósito especial y propósito general. Las ocho primeras posiciones del banco 0 soportan registros específicos.

Registros Especiales del área de datos.

Las ocho primeras posiciones del área de datos están reservadas para alojar registros de propósito especial, quedando las restantes libres para contener los datos u operandos que se desee (registros de propósito general). Figura 4.13.

El registro INDF que ocupa la posición 0 no está implementado físicamente y se le referencia en el direccionamiento indirecto de datos aunque se utiliza el contenido de FSR.

En la dirección 1 está el registro TMRO (Temporizador), que puede ser leído y escrito como cualquier otro registro. Puede incrementar su valor con una señal externa aplicada a la patita TOCKI o mediante el oscilador interno.

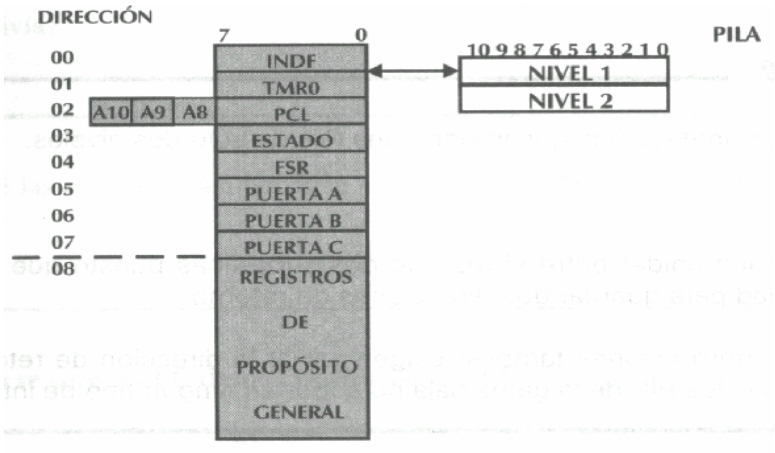


Figura 4.13 – Designación y ubicación de los ocho registros especiales.

El PC ocupa la posición 2 del área de datos en donde se halla el registro PCL al que se añaden 3 bits auxiliares y se conecta con los dos niveles de la Pila en las instrucciones CALL y RETLW.

El Registro de Estado ocupa la posición 3 y entre sus bits se encuentran los señaladores C, DC y Z y los bits PA1 y PAO que seleccionan la página en la memoria de programa. El bit 7 (PA2) no está implementado en los PIC de la gama baja.

FSR se ubica en la dirección 4 y puede usarse para contener la dirección del dato en las instrucciones con direccionamiento indirecto y también para guardar operandos en sus 5 bits de menos peso.

Los registros que ocupan las posiciones 5, 6 y 7 soportan las Puertas A, B y C de E/S. Pueden ser leídos y escritos como cualquier otro registro y manejan los valores de los bits que entran y salen por las patitas de E/S del microcontrolador.

La pila consta de un par de registros con 11 bits válidos que funcionan en forma de memoria LIFO (último en entrar, primero en salir). El contenido del PC se carga en el primer nivel de la Pila con la instrucción CALL y sucede lo contrario con la instrucción RETLW.

4.1.1.4. Arquitectura general y organización de la memoria en la gama media.

Novedades y aportaciones.

Todos los componentes de la gama media responden a la misma arquitectura básica del procesador interno y de la organización de la memoria y disponen del mismo repertorio de instrucciones. Sus diferencias estriban en la capacidad de la memoria y la disponibilidad de diferentes periféricos. Con carácter general, las aportaciones realizadas en esta gama con respecto a la anterior son las siguientes:

1 °) La longitud del código OP de las instrucciones pasa a ser de 14 bits frente a los 12 de los PICl6C5X. Este incremento permite manejar memorias de programa de 4 k palabras de 14 bits y memorias de datos de hasta 192 bytes frente a los 32 que permitían los microcontroladores de la gama baja.

2°) En la gama media existe la posibilidad de atender interrupciones, tan necesarias en las aplicaciones en tiempo real. El vector de interrupción se halla ubicado en 1a, dirección 0004 h de la memoria de programa.

3°) La Pila aumenta su profundidad hasta ocho niveles.

4°) Se añaden cuatro nuevas instrucciones al repertorio de la gama baja: RETURN, RETFIE, ADDLW y SUBLW. Dos instrucciones, OPTION y TRIS, se modifican para mantener la compatibilidad.

5°) Se ha redefinido la organización de la memoria de datos en bancos de 128 bytes cada uno.

6°) Se modifica el Registro de estado.

7°) El vector de reset se cambia a la posición 0000 h.

8°) Varía la actuación del reset sobre todos los registros, admitiéndose hasta cinco tipos diferentes de reset.

9°) Es posible salir del modo de Reposo (SLEEP) mediante interrupción.

10°) Se añade el registro PCLATH para contener los bits de más peso del PC, permitiendo manejar la paginación de la memoria. Este cambio obliga a redefinir la misión de los bits PA2, PA1 y PAO del Registro de estado.

11°) Los registros OPTION y TRIS pueden ser direccionados.

12°) Se incluyen dos temporizadores independientes, el que controla el "start-up" y el de "power-up ".

13°) Las líneas de la Puerta B pueden programarse para disponer de carga "pull-up" interna si actúan como entradas y también pueden provocar interrupciones.

14°) La patita RTCC puede actuar en algunos modelos como una línea más de la Puerta A (RA4/TOCKI).

15°) El registro FSR actúa completo.

16°) Se hace posible la programación del chip dentro del sistema puesto que para esta operación sólo se necesitan cinco patitas: VDD, VSS, VPP. RB6 (reloj) y RB7 (Entrada o salida del dato).

17°) Surge un nuevo registro denominado PCON, que tiene dos bits para diferenciar si el reset se ha producido por un fallo en la alimentación (Brown-Out) o por conexión la alimentación (Power-On-Reset).

18ª) Se ha mejorado el sistema de protección de código para actuar sólo sobre las zonas concretas del programa que se seleccionen.

19ª) Para convertir el código utilizado en los microcontroladores de la gama baja en código ejecutable para la gama media, se deben tener en cuenta las siguientes comprobaciones:

- Se deben modificar las operaciones que seleccionaban una página de la memoria en las instrucciones CALL y GOTO, revisando los valores de los bits PA2, PA1 y PAO.
- Revisar las operaciones de salto que escribían sobre el PC o le añadían un valor para asegurarse que se utilizan las páginas correctas.
- Eliminar los saltos entre los bancos de la memoria de datos.
- Verificar todas las escrituras sobre el Registro de Estado, OPTION y FSR dada su nueva estructura.
- Cambiar el vector de reset a la posición 0000 h.

Arquitectura interna y organización de la memoria.

Para soportar los nuevos recursos de la gama media, Microchip tuvo que ampliar y mejorar la estructura de la UCP y de la memoria, aunque manteniendo la arquitectura Harvard y el concepto RISC. También tuvo que añadir instrucciones y modificar alguna de las existentes.

En la figura 4.14 se presenta el esquema general al que responde la arquitecta básica de los procesadores PICl6CXX, similar a los de toda la gama. Apréciese que la memoria de datos está organizada en dos bancos con 128 posiciones cada uno como máximo. La mayoría de los registros específicos se ubican en las primeras posiciones de los bancos. En la memoria de datos de la figura los registros de propósito general ocupan las posiciones comprendidas entre la 20 h y la 6F h del banco 0. Todas las posiciones con trama no están implementadas en este caso concreto. A través de los registros y transfiriendo la información por el bus de datos se controlan los periféricos que comunican con el mundo exterior por las patitas de la Puerta A y la Puerta B.

Las líneas de E/S de las dos Puertas se corresponden con 13 patitas, siendo comunes las restantes cinco patitas en todos los modelos de 18 patitas, que se destinan a la alimentación, el oscilador y el reset.

En el esquema de la figura 4.14 se dibujan las memorias de datos y programa con la máxima capacidad que pueden alcanzar en la gama media. Las posiciones de la memoria de datos son de 1 byte y las de la memoria de programa, de 14 bits. Para direccionar los datos hacen falta 7 bits para elegir una posición del banco y 2 bits más para seleccionar el banco, ya que pueden existir hasta cuatro, aunque en esta gama sólo se usan dos en los modelos actuales.

El PC. Direccionamiento del programa.

El PC consta de 13 bits con los que se puede direccionar una memoria de código con una capacidad de hasta 8 k palabras de 14 bits cada una. La memoria se organiza en páginas de 2 k de tamaño.

El byte de menos peso del PC se corresponde con el contenido del registro PCL ubicado en la posición 02 h del banco 0. Los 5 bits de más peso del PC se corresponden con los 5 bits de menos peso del registro PCLATH en la posición 08 h del banco 0. Los bits de más peso del PC sólo se pueden escribir a través del registro PCLATH.
En las instrucciones de salto relativo, el resultado de la misma afecta sólo a lo bits de menos peso del PC. Los 5 bits de más peso se suministran desde PCLATH.

En las instrucciones GOTO y CALL los 11 bits de menos peso del PC se suministran desde el código OP. Los 2 bits de más peso del PC se cargan con los bits <4:3> del registro PCLATH. Como la memoria de programa se organiza en páginas de 2 k, la posición la seleccionan los 11 bits de menos peso, mientras que con los 2 bits de más peso del PC se elige la página.

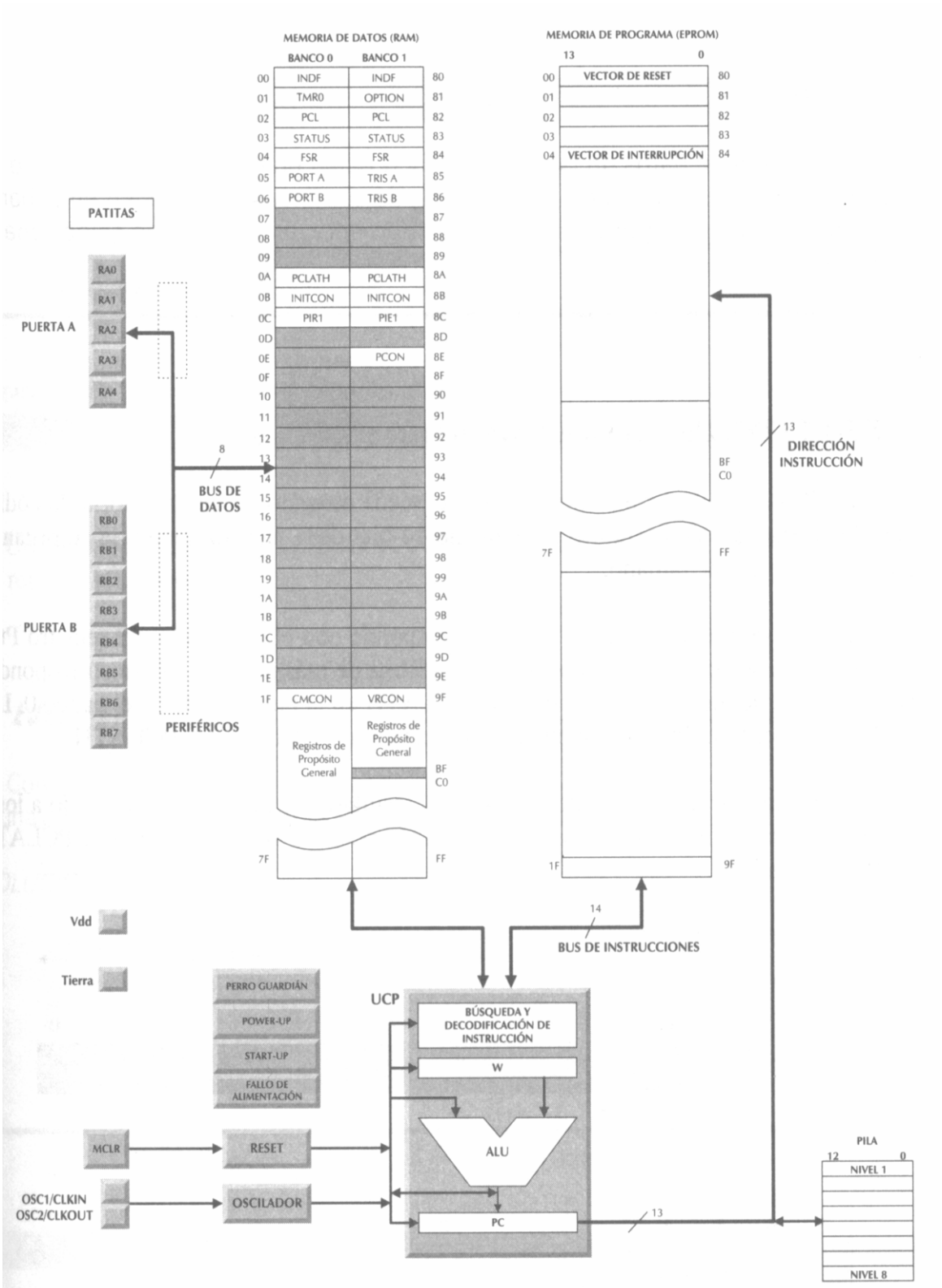


Figura 4.14 – Esquema general de la arquitectura básica de los PIC16CXX de la gama media, con 18 patitas en la cápsula.

Direccionamiento de los datos.

La memoria de datos en la gama media se organiza en un máximo de cuatro bancos, cada uno de los cuales puede constar de hasta 128 posiciones de tamaño byte.

Como los actuales dispositivos de esta gama no sobrepasan los 256 bytes, sólo se utilizan los bancos 0 y 1.

Para direccionar la memoria de datos que contiene los registros de propósito específico, y los de propósito general, existen dos modos de direccionamiento: directo e indirecto.

Ampliación del banco de registros.

La mayoría de los microcontroladores de la gama media disponen de un máximo de dos bancos de 128 x 8 posiciones, del máximo permitido que son cuatro bancos. El banco 0 comprende desde la posición 00 h a la 7F h y el banco 1, desde la 80 h a la FF h. Una vez elegido el banco, la dirección de la posición se determina con 7 bits.

Los registros especiales se ubican en las 32 primeras posiciones de cada banco. Los "registros de propósito general se ubican en trozos que quedan libres en ambos bancos. La mayoría de los registros son ya conocidos en la arquitectura de la gama baja aunque algunos tienen modificadas la función de sus bits. A continuación se describen las características de los principales registros de la gama media, que son similares a las del resto de componentes.

Registro de estado (STATUS).

Es un registro de 8 bits en el que se reflejan algunas circunstancias de interés sobre el estado del procesador. A continuación se indican la nomenclatura y la misión de los bits, los cuales pueden ser leídos y escritos, excepto /TO y /PD, que solo pueden leerse. Esta singularidad hay que tenerla en cuenta cuando se use este registro como destino en una instrucción.

R/W	R/W	R/W	R	R	R/W	R/W	R/W
IRP	RP1	RP0	/TO	/PD	Z	DC	C

C: Acarreo en el 8º bit.

1 = Acarreo en la suma y no en la resta. 0 = Acarreo en la resta y no en la suma.

DC: Acarreo en el 4º bit de menor peso.

1 = Acarreo en la suma y no en la resta. 0 = No acarreo en la suma. En la resta es al contrario respecto a la llevada.

Z: Zero.

1 = El resultado de una operación es 0. 0 = El resultado es distinto de 0.

/PD: Power Down.

1 = Tras conectar VDD o ejecutar "CLRWDT". 0 = Tras ejecutar "SLEEP".

/TO: Timer Out.

1 = Tras conectar VDD o ejecutar "CLRWDT" o "SLEEP".

0 = Al rebasar el tiempo de WDT.

RP1:RP0: Selección de los bancos de la memoria de datos.

00 = Banco 0 (00 – 7Fh).

01 = Banco 1 (80 - FFh).

10 = Banco 2 (100 – 17Fh).

11 = Banco 3 (180 – 1FFh).

IRP: Selección de bancos para el direccionamiento indirecto.

0 = Bancos 0 y 1 (00 - FFh).

1 = Bancos 2 y 3 (100 – 1FFh).

Figura 4.15 – Distribución y misión de los bits del Registro de estado.

Registro de opciones (OPTION).

A continuación se presenta la misión de los bits de este registro que ocupa la posición 81 h del banco 1. Con relación al de la gama baja, este añade los bits INTEDG, que regula el flanco activo para la generación de interrupción, y el RBPU, que conecta cargas Pull-up para las líneas de la Puerta B.

R/W	R/W	R/W	R	R	R/W	R/W	R/W
RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

RBPU: Conexión de cargas Pull-up para la puerta B.

1 = Todas las cargas Pull-up están desconectadas.

INTEDG: Tipo de flanco para la interrupción.

1 = RB0/INT sensible al flanco ascendente.

0 = RB0/INT sensible al flanco descendente.

T0CS: Fuente de reloj para TMR0.

1 = Pulsos introducidos por T0CKI (contador).

0 = Pulsos de reloj interno Fosc/4 (temporizador).

T0SE: Tipo de flanco activo del T0CKI.

1 = Incremento de TMR0 cada flanco ascendente.

0 = Incremento de TMR0 cada flanco descendente.

PSA: Asignación del divisor de frecuencia.

1 = Se le asigna al WDT.

0 = Se le asigna al TMR0.

PS2:PS0: Valor del divisor de frecuencia.

PS2	PS1	PS0	División del TMR0	División del WDT
0	0	0	1:2	1:1
0	0	1	1:4	1:2
0	1	0	1:8	1:4
0	1	1	1:16	1:8
1	0	0	1:32	1:16
1	0	1	1:64	1:32
1	1	0	1:128	1:64
1	1	1	1:256	1:128

Registro de interrupciones (INTCON).

Puesto que los microcontroladores PIC de la gama media admiten interrupciones, requieren un registro encargado de su regulación. La operatividad de sus bits se entenderá mejor cuando se explique la operatividad de las interrupciones. A continuación se muestra la estructura y la misión de los bits del registro INTCON. Algunos bits actúan como señalizadores del estado de un módulo y otros como bits de permiso o autorización para que se pueda generar la interrupción.

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF

GIE: Activación global de interrupciones.

1 = Concedido el permiso de interrupciones.

0 = Cancelado el permiso de interrupciones.

PEIE: Activación de la interrupción de periféricos.

1 = Activada. 0 = Desactivada.

T0IE: Activación de la interrupción del TMR0.

1 = Activada. 0 = Desactivada.

INTE: Activación de la interrupción externa.

1 = Activada. 0 = Desactivada.

RBIE: Activación de la interrupción de la puerta B.

1 = Interrupción activada. 0 = Interrupción desactivada.

T0IF: Señalizador de rebosamiento del TMR0.

1 = TMR0 ha rebosado. Se borra por software.

0 = El TMR0 no ha rebosado.

INTF: Software de estado de la interrupción externa.

1 = La entrada de interrupción se ha activado. Se borra por software.

0 = No hay interrupción externa.

RBIF: Señalizador de estado de la puerta B.

1 = Cambio de estado en cualquier línea de PB(RB<7:4>). Se borra por software.

0 = Ninguna entrada de PB ha cambiado.

Otros registros especiales.

- **PIE1:** Dispone de un bit que sirve para activar la interrupción provocada por el comparador.
- **PIR1:** También dispone de un bit funcional que actúa como señalizador sobre el cambio en la entradas del comparador. Se borra por software.
- **PCON:** Contiene dos bits con los que se puede diferenciar cuando el reset se ha originado por un fallo en la alimentación (BO) o por conexión de la misma (POR).

Palabras de configuración e identificación.

La Palabra de configuración en los PIC de la gama media se compone de 14 bits que se escriben durante el proceso de grabación del dispositivo. Dichos bits ocupan la posición reservada de la memoria de programa 2007 h. A continuación se muestra la estructura de la palabra de configuración.

13	12	11	10	9	8	7	6	5	4	3	2	1	0
CP1	CP0	CP1	CP0	CP1	CP0	-	BODEN	CP1	CP0	PWRTE	WDTE	Fosc1	Fosc2

Fosc <1:0>: Selección tipo de oscilador.

11 = Oscilador RC. 10 = Oscilador HS.

01 = Oscilador XT. 00 = Oscilador LP.

WDTE: Activación del WDT.

1 = WDT activado. 0 = WDT desactivado.

PWRTE: Activación del temporizador "Power-Up".

1 = Desactivado. 0 = Activado.

BODEN: Detección del "Brown-Out" (Fallo de alimentación).

1 = Detección activada. 0 = Detección desactivada.

CP <1:0>: Bits de protección de código.

CP<1:0> = 11 -> Off. CP<1:0> = 10 -> Off.

CP<1:0> = 01 -> ½ alta On. CP<1:0> = 00 -> Todo en On.

4.1.1.5. Puertas de entrada - salida.

Un recurso imprescindible en los microcontroladores es el que soporta las Entradas y Salidas con los periféricos del mundo exterior. Los PICl6C5X sólo disponen de líneas de E/S digitales, cada una de las cuales tiene su correspondencia con una patita de la cápsula. Dichas líneas se agrupan en tres puertas: Puerta A, Puerta B y Puerta C.

La Puerta A actúa lo mismo que un registro de E/S de lectura y escritura. S6lo son válidos los 4 bits de menos peso que corresponden con las patitas RA3:RAO. Los 4 bits de más peso no están implementados y cuando se leen siempre valen 0.

La Puerta B funciona como un registro de E/S de 8 bits bidireccionales, siendo accesibles todos ellos por las patitas RB7:RBO.

En los modelos PICl5C55/57, con 28 patitas, la Puerta C funciona como un registro de E/S de 8 bits. En los PICl6C54/CR54/C56, este registro se comporta como uno de propósito general al no existir suficientes patitas en las cápsulas para soportar estas líneas de E/S.

Los bits de cada puerta se configuran mediante los bits correspondientes de un registro de control asociado que recibe el nombre de TRIS. En realidad cada puerta soporta dos registros :

- El Registro de datos, al que se denomina Puerta A, B o C.
- El Registro de control TRISX, con el que se programa el sentido (Entrada o Salida) de las líneas de la Puerta X.

Las Puertas A, B y C se corresponden con las posiciones 5, 6 y 7 del área de datos. Cada uno de sus bits puede programarse como una línea de Entrada o de Salida, según se ponga un 1 ó un 0, respectivamente, en el bit del registro de control TRIS correspondiente. Figura 4.16.

Cada línea de E/S de las puertas se programa de forma independiente y puede ser Entrada o Salida. Cuando se produce un reset, todos los bits de los registros TRIS pasan a tener el valor 1 y todas las líneas de E/S actúan como Entrada por evidentes motivos de seguridad para evitar daños irreparables.

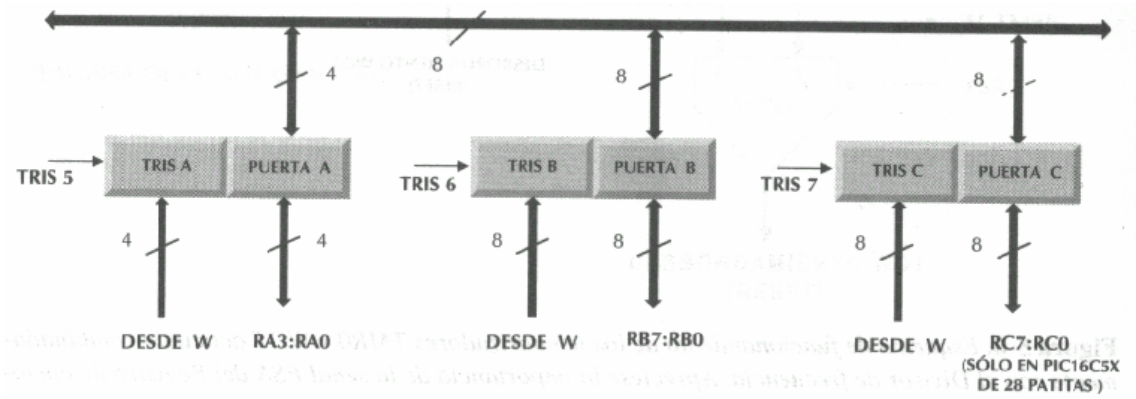


Figura 4.16 – Cada puerta de E/S tiene asociado un registro TRIS que configura como Entrada o Salida cada línea.

La información presente en una línea de Entrada se muestrea al iniciarse un instrucción y debe mantenerse estable durante ese tiempo.

Como ya indicamos cada línea de una puerta puede suministrar una corriente máxima de 20 mA actuando como salida y absorber hasta 25 mA cuando actúa como entrada. Sin embargo, hay que tener en cuenta que existen unas limitaciones de disipación de potencia en el chip que son:

a) La Puerta A puede absorber un máximo de 80 mA y suministrar un máximo de 50 mA en total.
b) La Puerta B puede absorber un máximo de 150 mA y suministrar entre todas sus líneas un máximo de 100 mA.

Estas restricciones obligan a limitar la corriente de salida total de cada puerta así como la de entrada. Por lo tanto, habrá que conjugar los máximos admitidos por cada línea con los máximos permitidos por cada puerta, que comprende a todas las líneas.

Las puertas de entrada se leen mediante la instrucción movf PORTX, W siendo válido el dato sólo en el momento de la lectura pues no se memoriza.

Las puertas de salida se escriben con la instrucción movwf PORTX .

Es posible modificar el valor de algún bit particular de una puerta mediante las instrucciones bsf (poner a 1) y bcf (poner a 0). También se puede leer el valor de un bit de una puerta con las instrucciones btfss y btfsc.

Las puertas que contienen entradas y salidas necesitan una atención especial al escribir el programa. Instrucciones como bsf y bcf comienzan leyendo el valor de la puerta y cargándolo en el registro W; allí ejecutan la puesta a 1 ó a 0 del bit seleccionado y, luego, depositan el registro W en la puerta. También hay que tener en cuenta las modificaciones que se produzcan en las patitas que son entrada y pasan a salida, pues pueden estar presentes datos antiguos en el registro de salida de la puerta al ser memorizados.

Hay que prestar mucha atención a las operaciones que, tras una lectura de una puerta, sigue una escritura de la misma. Se debe dejar pasar un tiempo determinado para que se estabilice el voltaje de las patitas. Insertando entre la lectura y la escritura una instrucción NOP o cualquier otra que no implique a las puertas, se eliminan estos errores potenciales.

Conexiones de entrada – salida.

En el diseño del generador el microcontrolador posee conexiones tanto de entrada como de salida, y estas se describen a continuación:

Conexiones de salida: Se configuran como salidas las líneas 0, 2, 3 y 4 de la Puerta B, que se corresponden con las señales de sincronismo compuesto y componentes de color respectivamente. Los circuitos asociados a estas salidas se describirán en el apartado correspondiente al conversor de vídeo, ya que dichos circuitos adaptan las señales para el conversor.

Conexiones de entrada: Las conexiones de entrada se corresponden con las líneas número 2 y 3 de la Puerta A. La línea número dos está destinada al control del teclado, mientras que la número tres se reserva para conservar la compatibilidad en caso de utilizarse otro software distinto del original (por ejemplo software de vídeo-aficionados disponible en la red).

El teclado consiste básicamente en un pulsador y una resistencia de polarización tal como se muestra en la figura 4.17.

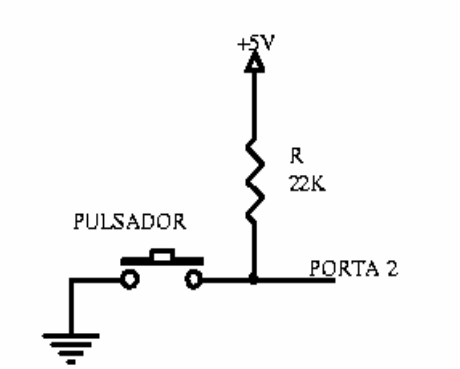


Figura 4.17 – Circuito del teclado en el generador de vídeo.

Mientras no se apriete el pulsador la resistencia de polarización mantiene un nivel lógico alto en la entrada del microcontrolador. Cuando se aprieta el pulsador, se cierra el circuito y la entrada del microcontrolador pasa a nivel bajo, evitando la resistencia de polarización que se produzca un cortocircuito.

El valor de la resistencia de polarización coincide aproximadamente con el de las resistencias de pull-up que integra el microcontrolador, garantizándose de esta forma una respuesta estable.

El software comprueba el estado de PORTA2 al final del campo par, es decir, al final de cada cuadro. Si el nivel encontrado es un nivel alto se continua barriendo un nuevo cuadro, pero si el nivel encontrado es un nivel bajo (se ha pulsado para solicitar un cambio de modo) se salta al siguiente modo de funcionamiento.

En la figura 4.18 se muestra la sección código que se encarga del control del teclado. El código se ha escrito en lenguaje ensamblador debido a la necesidad de una temporización extremadamente exacta en las funciones encargadas de generar las señales de sincronismo y vídeo.

El software de control sigue los siguientes pasos:

1) Lee el estado de PORTA2. Si se encuentra un nivel lógico alto (no se ha pulsado) se continua el barrido de un nuevo cuadro, mientras que si se encuentra un nivel lógico bajo (se ha pulsado) se interrumpe el bucle de barrido.
2) Una vez interrumpido el barrido espera unos 20ms aproximadamente a que se estabilice la señal (evitamos los rebotes de contacto), y a continuación espera a que se suelte el pulsador (nivel alto en la entrada). De esta forma se evitan saltos a un modo inesperado (se asegura la secuencia de modos).
3) Por ultimo, se salta al siguiente modo de funcionamiento actualizándose los bits señalizadores de modo (banderas de modo).

```
leetec      btfsc   porta,2         ;lee bit 2 porta -> si es 0 salta
            return                  ;retorna
            clrf    c1              ;
            movlw   d'65'           ;
            movwf   c2              ;
delay       decfsz  c1,f            ;retardo
            goto    delay           ;de 20ms aprox.
            decfsz  c2,f            ;para eliminar los
            goto    delay           ;rebotes de contacto
leetec2     btfss   porta,2         ;lee bit 2 porta -> si es 1 salta
            goto    leetec2         ;                -> si es 0 espera
actmod      btfsc   flag,cb         ;si modo 1 -> modo 2
            goto    iniraster1      ;salta a raster1
            btfsc   flag,ra1        ;si modo 2 -> modo 3
            goto    iniraster2      ;salta a raster2
            btfsc   flag,ra2        ;si modo 3 -> modo 4
            goto    iniraster3      ;salta a raster3
            btfsc   flag,ra3        ;si modo 4 -> modo 5
            goto    iniraster4      ;salta a raster4
            btfsc   flag,ra4        ;si modo 5 -> modo 6
            goto    inidamero       ;salta a damero
            btfsc   flag,dam        ;si modo 6 -> modo 7
            goto    inirejilla      ;salta a rejilla
            btfsc   flag,rej        ;si modo 7 -> modo 1
            goto    inicolbar       ;salta a colbar
```

Figura 4.18 – Software de control del teclado en el generador de vídeo.

4.1.2. CONVERSOR DE VÍDEO CXA 1645.

4.1.2.1. Descripción general.

El circuito integrado CXA1645P/M de SONY es un codificador que convierte señales analógicas RGB en una señal de vídeo compuesto. El circuito integrado contiene gran parte de la circuiteria necesaria para la codificación. La señal de vídeo compuesto, y las señales de luminancia y crominancia para la salida de súper-vídeo, se obtienen directamente introduciendo las señales de entrada sincronismo compuesto, subportadora de color y componentes de color RGB.

El circuito es recomendado para aplicaciones de procesamiento de imagen en ordenadores personales y vídeo juegos.

Modo de operación.

Las señales RGB que entran por las patillas 2,3 y 4 pasan a través de un buffer de entrada y salen por las patillas 23, 22 y 21 respectivamente.

Un circuito matricial realiza operaciones en cada señal de entrada, generando la señal de luminancia Y, y las señales de diferencia de color R-Y y B-Y. La señal de luminancia pasa a través de una línea de retardo para ajustar el tiempo de retardo con la señal de color C.

Después de la adición de la señal de sincronismo compuesto CSYNC que entra por la patilla 10, la señal de luminancia Y sale por la patilla 16.

La subportadora de color que entra por la patilla 6, pasa al desplazador de fase donde la fase se desplaza 90º. Luego la subportadora entra en los moduladores y es modulada por las señales R-Y y B-Y. Las subportadoras moduladas son mezcladas y pasan por un filtro paso banda que elimina las componentes armónicas superiores, y finalmente sale por la patilla 15 como la señal de crominancia C.

Al mismo tiempo, las señales de luminancia y crominancia se mezclan y salen por la patilla 20 como la señal de vídeo compuesto.

Salva de Color.

El circuito integrado genera la salva de color con la temporización que se muestra a continuación y de acuerdo con la señal de sincronismo compuesto de entrada.

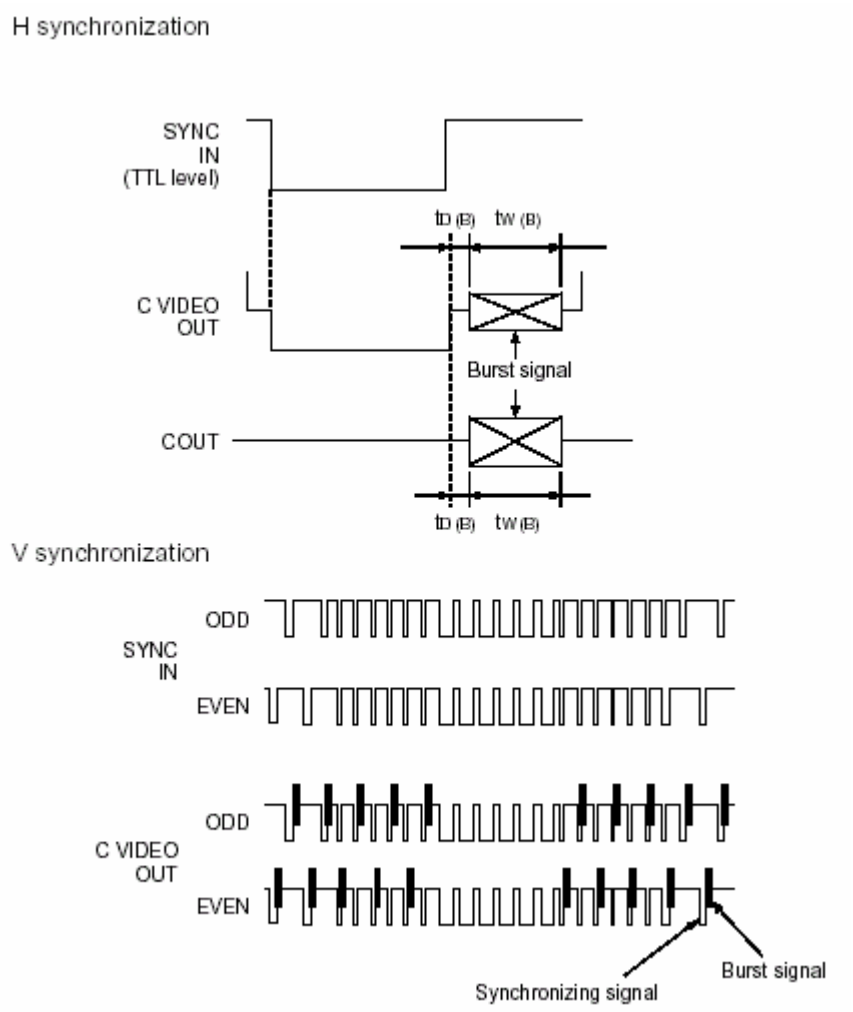


Figura 4.19 – Generación de la Salva de Color.

4.1.2.2. Características técnicas.

Las principales características proporcionadas por el fabricante son las siguientes:

Características principales:

- Alimentación simple de 5V.
- Compatible con los sistemas NTSC y PAL.
- Incluye drivers para 75Ω (salidas RGB, vídeo compuesto, luminancia y crominancia).
- Posibilidad de subportadora senoidal o rectangular.
- Incluye filtro paso-banda para la señal de crominancia, y línea de retardo para la señal de luminancia.
- Incluye circuito moduladores B-Y y R-Y.
- Incluye circuito de alternancia PAL.
- Circuito generador de salva de color.

Rangos máximos absolutos:

- Suministro de voltaje (Vcc).....14V
- Temperatura de operación (Topr).....-20 a 75 ºC
- Temperatura de almacenamiento (Tstg).....-65 a 150 ºC
- Disipación de potencia (PD).....CXA1645P – 1250 mW

Condiciones de operación recomendadas:

- Suministro de voltaje (Vcc1,2).....5.0 +/- 0.25 V

Tabla 4.2 - Descripción de terminales del conversor de vídeo CXA1645.

Nombre	Número	Voltaje	Descripción
GND1	1	0V	Tierra para todos los circuitos excepto para RGB, SVC, SSV.
RIN-GIN-BIN	2-3-4	Clamped 2V	Entradas RGB.
NC	5	-	No conectada.
SCIN	6	-	Entrada subportadora
NPIN	7	1.7V	NTSC/PAL
BFOUT	8	H:3.6V L:3.2V	Pulso BF.
YCLPC	9	2.5V	Temporización para clamp.
SYNC IN	10	2.2V	Entrada sincronismo compuesto.
VCC1	12	5V	Alimentación excepto para los circuitos RGB, SY, SC.
IREF	13	2V	Referencia interna de corriente.
VREF	14	4V	Referencia interna de voltaje.
COUT	15	2.2V	Salida señal crominancia.
YOUT	16	1.3V Nivel negro	Salida señal luminancia.
YTRAP	17	1.6V Nivel negro	Filtro de color.
FO	18	2V	Ajuste de frecuencia interna.
VCC2	19	5V	Alimentación para los circuitos RGB, SY, SC.
CVOUT	20	1.2V Nivel negro	Salida de vídeo compuesto.
BOUT- GOUT-ROUT	21-22-23	1.7V Nivel negro	Salidas RGB.
GND2	24	0V	Tierra para los circuitos RGB, SY, SC.

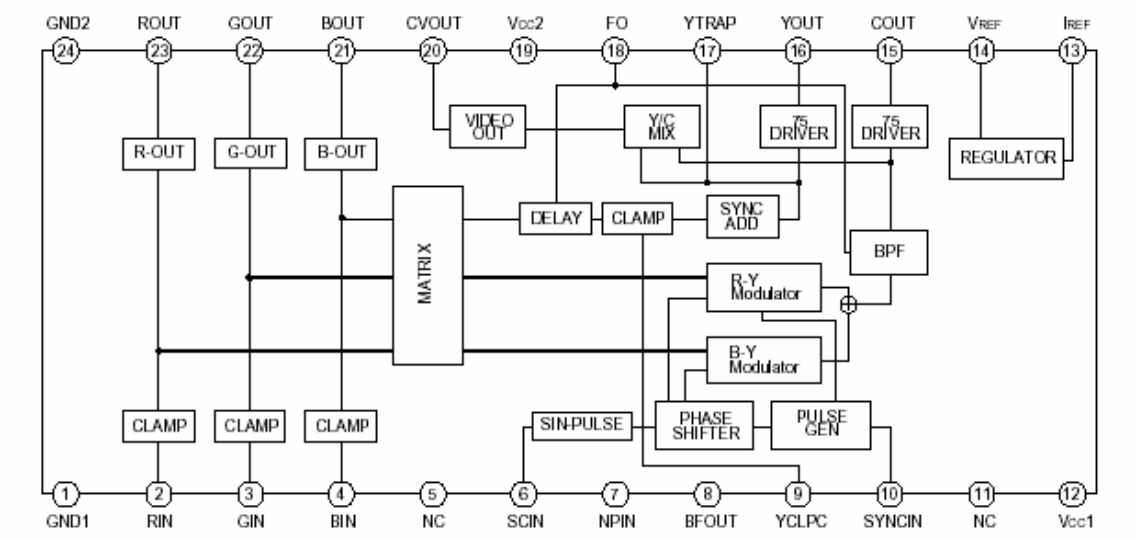


Figura 4.20 – Diagrama de terminales del conversor CXA1645.

4.1.2.3. Conexiones de entrada-salida.

Como conexiones de entrada-salida se tienen las siguientes:

Circuitería externa: Forman parte de la circuitería externa aquellos componentes que, por problemas de integración o bien por necesidad de manipulación, han de situarse fuera da la cápsula del circuito integrado. Estos componentes externos son mínimos y se enumeran a continuación:

- Condensador de 100nF entre patilla 9 y tierra: Determina la constante de tiempo para la conexión de la señal de luminancia.
- Condensador de 100nF // resistencia de 47KΩ entre patilla 13 y tierra: Determina la corriente interna de referencia.
- Condensador de 10uF entre patilla 14 y tierra: Filtra el voltaje interno de referencia.
- Resistencia de 16KΩ entre patilla 18 y tierra: Ajusta la frecuencia fo del filtro paso-banda interno.

Conexiones de entrada: Las conexiones de entrada están constituidas por una serie de redes que adaptan las señales provenientes del microcontrolador y el oscilador de croma a los requerimientos de entrada del conversor. Estas redes se describen y calculan a continuación:

- **Adaptación de entradas RGB:** Convierten los niveles de salida TTL de las señales RGB que proporciona el microcontrolador, en niveles de 1Vpp que necesita el conversor para reproducir colores con un 100% de saturación. La red encargada de dicha función se muestra a continuación en la figura 4.21, y consiste en un divisor de tensión resistivo, junto con un condensador para el desacoplo de la señal.

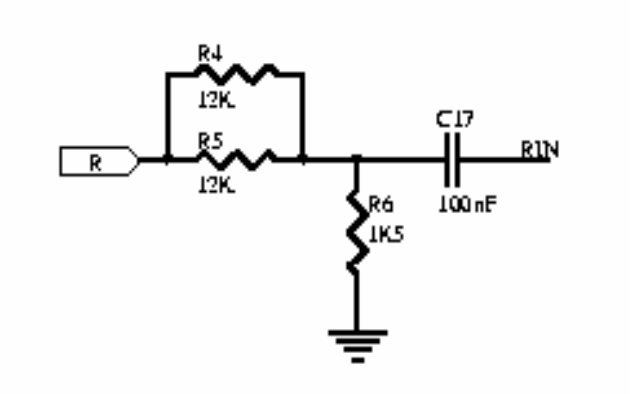


Figura 4.21 – Red de adaptación para las señales RGB.

Los requerimientos del diseño son:

- Tensión de entrada de 1Vpp.
- Impedancia de entrada lo más baja posible

Y como limitaciones se tiene:

- Tensión de salida del microcontrolador de 5V.
- Intensidad de salida del microcontrolador de 20mA por patilla, y 100mA en total por la puerta B.

Los valores para los componentes son los que se muestran en la figura, es decir, R4 y R5 = 12KΩ, R6 = 1.5KΩ, y C17 = 100nF.

El valor de la capacidad de entrada es el recomendado por el fabricante, y se supone que ofrece una impedancia despreciable a la frecuencia de trabajo.

Con esta consideración y los valores de componentes expuestos anteriormente, los resultados obtenidos son los siguientes:

- **Zin** ≈ 6KΩ // 1.5KΩ = 1.2KΩ.
- **Vin** ≈ (5V * 1.5KΩ) / 7.5KΩ = 1V.
- **Io** ≈ 5V / 7.5KΩ = 0.66mA.

Tal como se puede observar en los cálculos anteriores, se cumplen todos los requerimientos iniciales.

- Adaptación de entrada SYNC: La adaptación consiste en un simple filtrado paso-bajo de la señal de sincronismo compuesto que proporciona el microcontrolador, para de esta forma eliminar las componentes de alta frecuencia que podrían provocar un mal funcionamiento del circuito integrado conversor de vídeo. El circuito se muestra en la figura 4.22, siendo la frecuencia de corte del filtro para los valores recomendados por el fabricante:

$$fo = Wo / (2\pi) = 1 / (2\pi RC) = 1/ (6.28 * 2.2K\Omega * 47pF) = 1.54MHz.$$

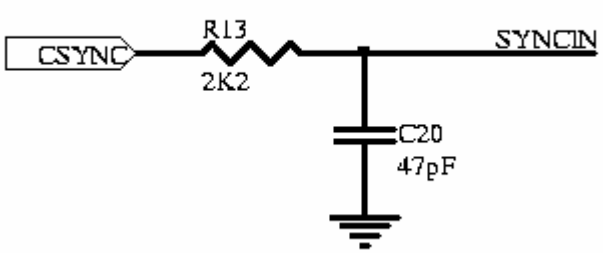


Figura 4.22 – Red de adaptación para la señales de sincronismo compuesto.

- Adaptación de entrada SC: La adaptación también consiste en un filtrado paso-bajo de la señal, para de esta forma eliminar las componentes de alta frecuencia que podrían provocar un mal funcionamiento del conversor de vídeo. En este caso la señal de entrada es la subportadora de color, la cual es generada por un oscilador se describirá seguidamente. El circuito de filtrado se muestra en la figura 4.23, siendo la frecuencia de corte del filtro, para los valores más próximos a los recomendados por el fabricante:

$$fo = Wo / (2\pi) = 1 / (2\pi RC) = 1/ (6.28 * 2.2K\Omega * 4,7pF) = 15.4MHz.$$

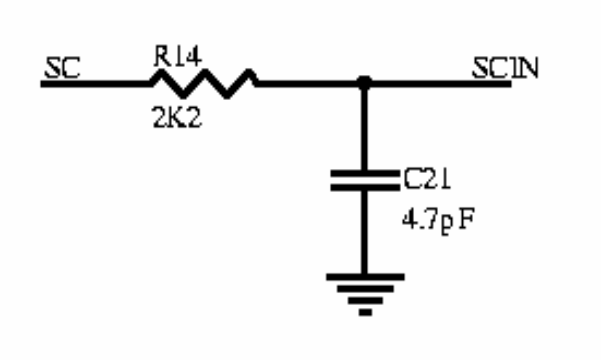


Figura 4.23 – Red de adaptación para la subportadora de color.

El oscilador que genera la subportadora de color está construido con inversores CMOS de alta velocidad, exactamente con el circuito integrado 74HC04N. EL circuito integrado contiene seis unidades inversoras, de las cuales solo se utilizan tres, tal como puede observarse en el esquema del circuito representado en la figura 4.24.

La resistencia R2 forma parte de una red RC que determina la constante de tiempo del oscilador, donde el condensador de dicha red se sustituye por el cristal de cuarzo XT2 para proporcionar una mayor estabilidad de la frecuencia de trabajo.

La frecuencia fo del cristal XT2 gobierna con precisión el periodo de la onda cuadrada, y el condensador variable C16 ayuda en el ajuste del mismo.

El condensador C15 se elige para que con fo tenga una reactancia despreciable, y este junto con la resistencia R3 ayudan a suprimir los armónicos superiores.

El ultimo inversor no forma parte del oscilador, y sirve solo para adaptar la impedancia de salida.

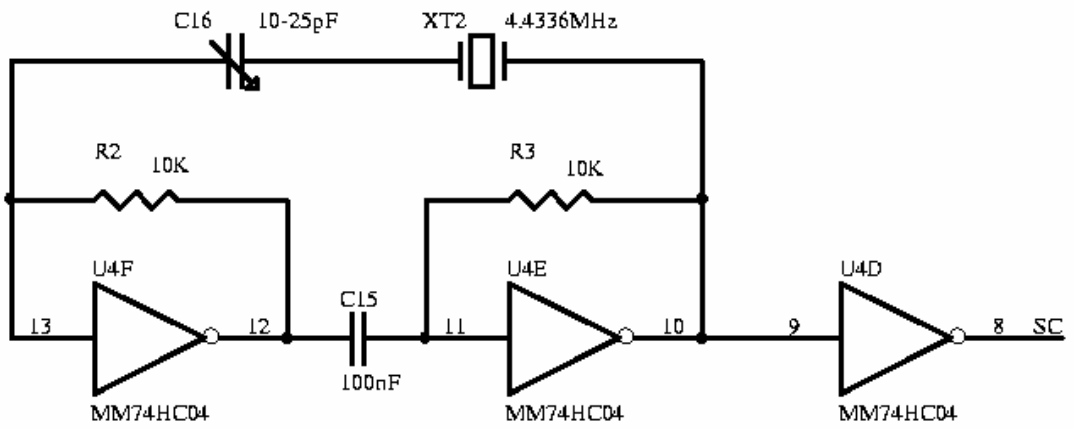


Figura 4.24 – Circuito del oscilador de la subportadora de color.

Conexiones de salida: Las conexiones de salida constituyen el ultimo paso de la señal antes de salir al exterior, y sus funciones son las de filtrado y adaptación de impedancias. Estos circuitos de salida son los que se muestran a continuación:

- **Filtro YTRAP:** Consiste en un filtro trampa que se conecta de manera opcional en la patilla número 17 del conversor de vídeo. Este filtro tiene como misión eliminar las componentes de frecuencia de la subportadora de color presentes en la señal de luminancia. El circuito se muestra en la figura 4.25, y para los valores especificados el filtro está centrado aproximadamente a la frecuencia de la subportadora de color (4.43MHz), siendo su valor exacto el dado por la ecuación:

$$fo = Wo / (2\pi) = 1 / (2\pi \sqrt{LC}) = 4.44MHz.$$

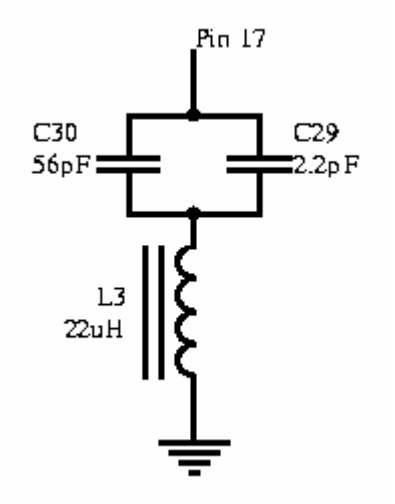


Figura 4.25 – Filtro trampa para la señal de luminancia.

- Circuitos de salida: El circuito es común para las tres salidas, es decir, para las salidas de vídeo compuesto, luminancia y crominancia, y su función es la de adaptar la impedancia de salida y de esta forma el nivel de salida.

El circuito se representa en la figura 4.26, y está constituido por dos resistencias de 150Ω en paralelo que dan lugar a la carga estándar de 75Ω.

El condensador de 220uF sirve para evitar la componente continua en la carga, ya que los circuitos de salida no tienen capacidad para alimentar a una carga de 75Ω con acoplamiento directo.

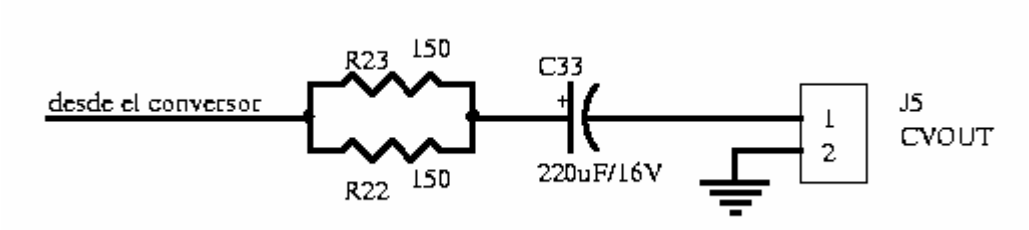


Figura 4.26 – Circuito que adapta las señales de salidas del conversor.

4.1.3. CONVERSOR DE VÍDEO MC1377.

4.1.3.1. Descripción general.

El circuito integrado MC1377 de MOTOROLA es un codificador de RGB a PAL para televisión en color. El circuito integrado puede generar una señal de vídeo compuesto a partir de entradas Roja, Verde, Azul, y Sincronismo. De las características que incluye el circuito se destacan:

- Un oscilador para la Subportadora de Color.
- Un desplazador de fase de 90º controlado por voltaje.
- Dos moduladores de color de doble banda con portadora suprimida (DSBSC).
- Una matriz de entrada RGB con conexión a nivel de negro.

Estas características permiten el diseño de sistemas con muy pocos componentes externos y en consecuencia, la ejecución de sistemas comparables con equipos de estudio con componentes externos comunes en sistemas receptores.

Otras características interesantes del circuito son:

- Posibilidad de oscilador de referencia interno o externo.
- Ajuste opcional de los ejes de color (nominal 90° +/- 5°).
- Compatible con sistemas PAL y NTSC.
- Regulador interno de 8.2V.

El modo de operación del circuito es similar al del anterior conversor, por lo que no es necesaria su explicación.

En comparación con el conversor de SONY (CXA1645), este solo posee salida de vídeo compuesto. Además no integra tanta circuitería, siendo necesario la utilización de circuitos de filtrado y retardo conectados en el exterior de la cápsula.

4.1.3.2. Características técnicas.

Las principales características proporcionadas por el fabricante son las siguientes:

Condiciones de operación máximas:

- Suministro de voltaje (Vcc).....15Vdc.
- Temperatura de almacenamiento (Tstg).....-65 a 150 ºC.
- Disipación de potencia de la cápsula (PD).....1.25W.
- Temperatura de operación (Ta).....0 a 70 ºC.

Condiciones de operación recomendadas:

- Suministro de voltaje.....12Vdc.
- Corriente IB.....0 a 10mA.
- Sincronismo, nivel de negro.....1.7 a 8.2 Vdc.
- Sincronismo, nivel típico.....-0.5 a 0.9Vdc.
- Sincronismo, ancho de pulso.....2.5 a 5.2 us.
- Amplitud de entradas RGB.....1Vpp.
- Niveles de pico RGB para acoplamiento directo.....2.2 a 4.4 V.
- Ancho de banda de la señal de crominancia.....0.5 a 2MHz.
- Nivel de subportadora externa.....0.5 a 1Vpp.

Tabla 4.3 - Descripción de terminales del conversor de vídeo MC1377.

Símbolo	Patilla	Descripción
tr	1	Ajuste constante de tiempo para disparo de salva.
Sync	2	Entrada de sincronismo compuesto.
R	3	Entrada de señal Roja. Zin = 10KΩ, Vin = 1Vpp.
G	4	Entrada de señal Verde. Zin = 10KΩ, Vin = 1Vpp.

Tabla 4.3 - Descripción de terminales del conversor de vídeo MC1377.

B	5	Entrada de señal Azul. Zin = 10KΩ, Vin = 1Vpp.
Yout	6	Salida de luminancia. Ajuste de tiempo de retardo.
Vclamp	7	Patilla de conexión a tierra.
Yin	8	Entrada de luminancia. Zin = 10KΩ.
CVout	9	Salida de vídeo compuesto. Zout = 50Ω.
Chroma in	10	Entrada de crominancia. Zin = 10KΩ.
B -Y Clamp	11	Conexión a tierra durante nivel de negro.
R – Y Clamp	12	Conexión a tierra durante nivel de negro.
Chroma out	13	Salida de crominancia. Zout = 50Ω.
VCC	14	Suministro de potencia. 12 +/- 2V, 35 mA.
GND	15	Conexión a tierra.
VB	16	Regulador interno de referencia. 8.2V, 10mA.
Oscin	17	Entrada del oscilador de referencia. 5KΩ.
Oscout	18	Salida del oscilador de referencia
Φm	19	Corrección del ángulo relativo entre R-Y y B-Y.
NTSC/PAL	20	Selección del modo de operación.

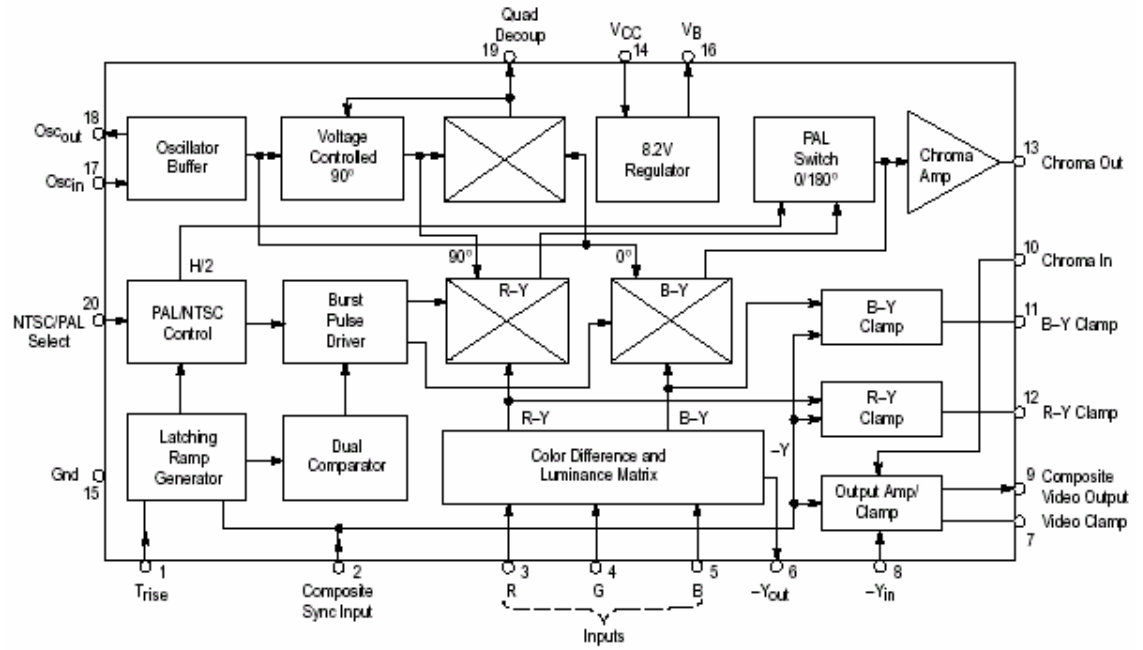


Figura 4.27 – Diagrama de terminales del conversor MC1377.

4.1.3.3. Conexiones de entrada-salida.

Como conexiones de entrada-salida se tienen las siguientes:

Circuitería externa: Del mismo modo que en el caso anterior, forman parte de la circuitería externa aquellos componentes que, por problemas de integración o bien por necesidad de manipulación, han de situarse fuera da la cápsula del circuito integrado. A diferencia del caso anterior, estos componentes externos son más numerosos, y estos se citan a continuación:

- **Filtro de color:** La mayoría de monitores y receptores tienen el ancho de banda de la FI de color limitado a aproximadamente +/- 0.5MHz, por lo que se recomienda que el circuito codificador tenga también limitado el ancho de banda a +/- 0.5MHz, lo cual se consigue mediante la inserción de un filtro paso banda entre los terminales 13 y 10.

Para un nivel adecuado de la señal de color en la salida de vídeo, se recomienda un ancho de banda de +/- 0.5MHz con unas perdidas de inserción en la mitad de banda de 3dB. Para implementar este filtro paso banda se recomienda un circuito con transformador de sintonía fija TOKO, el cual introduce un retardo de 350ns que produce un desplazamiento visible en la información de color, y blanco y negro. La solución consiste en colocar una línea de retardo en el camino de la señal de luminancia entre los terminales 6 y 8, para de esta forma realinear las dos componentes.

No obstante, en el caso de información RGB con muy baja resolución (menos de 1.5MHz), el circuito de reducción de ancho de banda que se requiere es más simple. El circuito que se muestra en la figura 4.28 suministra una pequeña reducción del ancho de banda, pero suficiente para eliminar el problema de alineación de las señales. El circuito presenta las siguientes características:

- Perdidas de inserción: 3dB.
- Ancho de banda: +/- 1MHz.
- Retardo: 100ns.

Las especificaciones del filtro se prueban en la simulación mostrada en la figura 4.29, en la cual se han tenido en cuenta además los niveles e impedancias de entrada/salida del circuito integrado.

Además, tal como se muestra en el circuito de la figura 4.28, se aprovecha la exteriorización del filtro para introducir una red capaz de eliminar la información de color. Esta funcionalidad nos permitirá disponer de todos los patrones de prueba en blanco y negro, y se consigue desviando la señal a tierra a través de un interruptor y un condensador. El condensador bloquea la componente continua evitando la posibilidad de un cortocircuito, y al mismo tiempo elimina eficazmente la información de color debido a su gran capacidad.

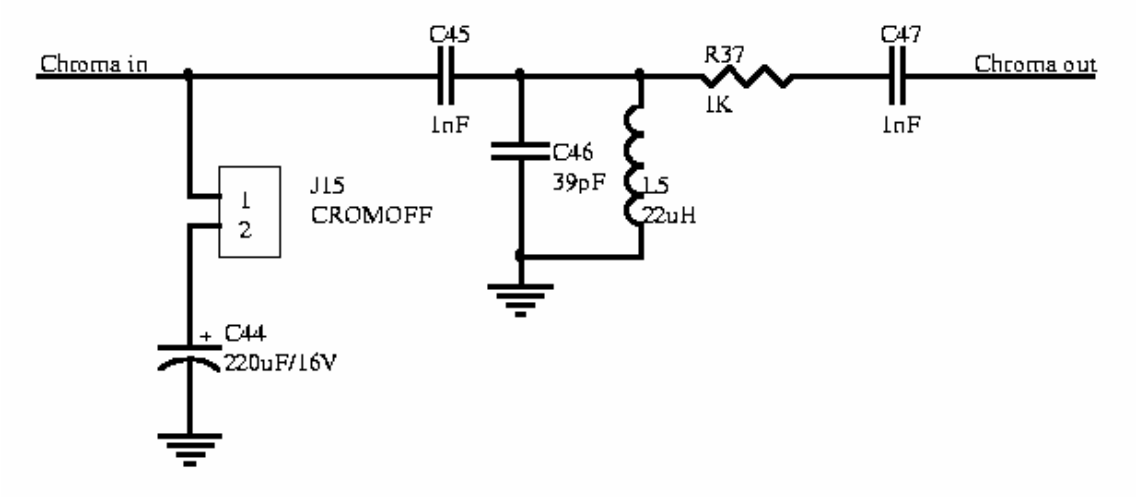


Figura 4.28 – Filtro de color externo para el conversor MC1377.

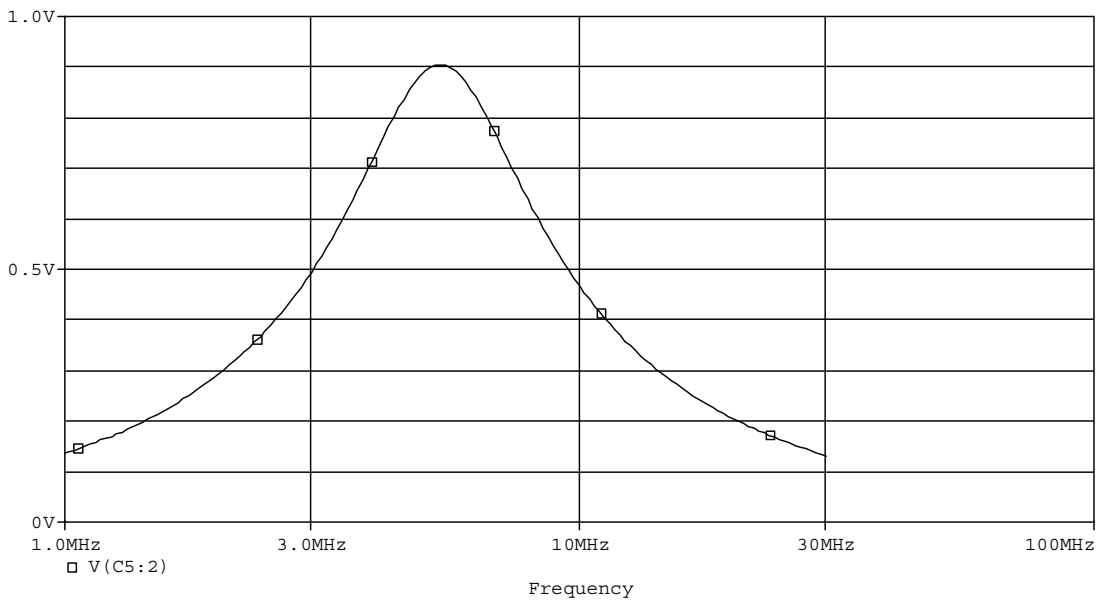


Figura 4.29 – Respuesta del filtro paso banda del filtro de color.

- Línea de retardo: Tal como se ha comentado en el apartado anterior, el filtro de color introduce un retardo que es preciso corregir. La solución consiste en introducir el mismo retardo en la línea de luminancia, lo cual se consigue con una línea de retardo. El modelo utilizado es el 11ACB10112E con encapsulado DIL de 14 pines. Consiste en una línea pasiva de retardo constante de elementos concentrados, la cual ofrece 10 derivaciones de retardo a espacios iguales con posibilidad de conexión en serie de varias líneas de retardo, ya que no se incluyen resistencias en las terminaciones. El retardo de una derivación es de 10ns, consiguiéndose un retardo total de 100ns con la conexión en serie de todos los elementos.

En el circuito de la figura 4.30 se muestra la línea de retardo. En ella pueden observarse unas resistencias de 1KΩ que adaptan el nivel de la señal, las cuales han de estar en concordancia con la impedancia característica de la línea de retardo.

Como en el caso anterior, se incluye una red que en este caso permite eliminar la señal de luminancia.

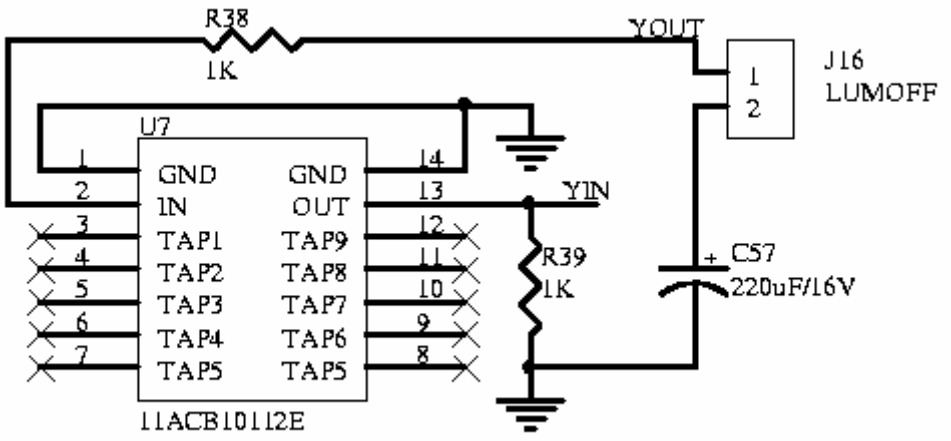


Figura 4.30 – Circuito de retardo para la línea de luminancia.

- Control de Salva de color: La salva de color se sitúa en el pórtico posterior del impulso de borrado horizontal. Para ello se genera una rampa de tensión y se compara el nivel de esta con unos niveles de referencia que nos indicaran los instantes de comienzo y fin, fijándose de esta forma el tamaño y la posición final de la salva en el pórtico. La rampa se genera a partir de la tensión de referencia disponible en la patilla 16, y para ello se utiliza una red RC en la que se incluye una resistencia variable que permite el control de la constante de tiempo de la rampa.

Los valores de los componentes se muestran en la figura 4.31, y se han elegido partiendo de las recomendaciones del fabricante, ya que de esta forma se garantiza la pendiente de la rampa y por tanto la duración de la salva.

También se incluye un interruptor que permite eliminar la salva de color, siendo esta función muy útil para probar el funcionamiento de los circuitos de color en receptores y monitores.

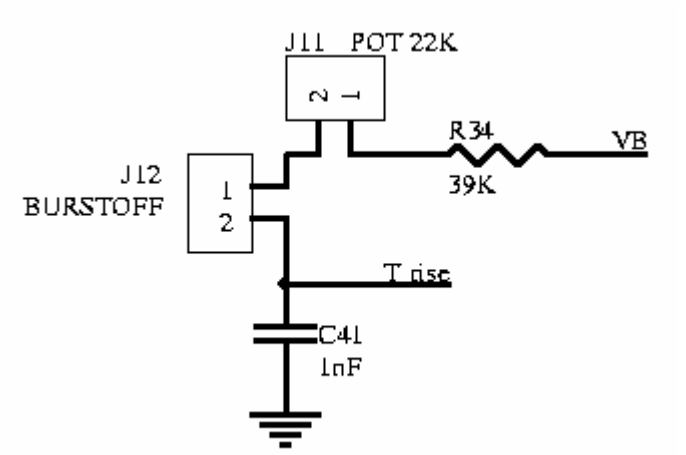


Figura 4.31 – Circuito de control para la salva de color.

- Circuitos auxiliares: Los circuitos auxiliares se corresponden con una serie de componentes que por motivos de integración y/o configuración han de situarse en el exterior de la cápsula. Estos componentes son:

- Condensador de 10nF entre patilla 7 y tierra: Sirve para la conexión a tierra de la señal de vídeo durante el nivel de negro.
- Condensadores de 100nF desde patillas 11 y 12 hasta tierra: Realizan la conexión a tierra de las señales B-Y y R-Y respectivamente durante el nivel de negro.
- Condensador de 100nF entre patilla 16 y tierra: Filtra la tensión de referencia presente en esa patilla.
- Condensadores de 220pF, cristal de 4.43MHz y condensador variable, entre patillas 17, 18 y tierra: Configuran el oscilador interno para generar la subportadora de color.

Conexiones de entrada: Las conexiones de entrada están constituidas por una serie de redes que adaptan las señales, provenientes del conversor CXA1645 y el microcontrolador, a los requerimientos de entrada del conversor MC1377. Estas redes se describen y calculan a continuación:

- Adaptación de entradas RGB: Convierten los niveles de las salidas RGB que proporciona el conversor CXA1645, en niveles de 1Vpp que necesita el conversor MC1377 para reproducir colores con un 100% de saturación. La red encargada de dicha función se muestra a continuación en la figura 4.31, y consiste en un divisor de tensión resistivo, unos condensador para el desacoplo de la señal, y una red que permite eliminar la señal de entrada.

La función de supresión de la señal de entrada nos permitirá:

1) La generación de muchos más patrones de prueba, lo cual se consigue con diferentes combinaciones de las señales de entrada.
2) Apreciar el efecto que provocaría la ausencia de una o varias componentes de color debido a un mal funcionamiento de los circuitos asociados a las mismas.

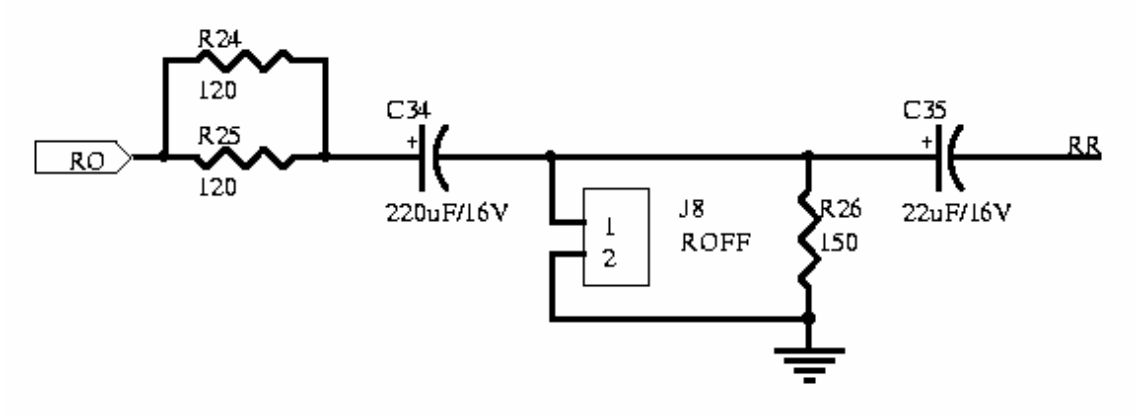


Figura 4.31 – Red de adaptación para las señales RGB.

Los requerimientos del diseño son:

- Tensión de entrada de 1Vpp.
- Impedancia de entrada lo más baja posible

Y como limitaciones se tiene:

- Tensión de salida del conversor CXA1645 de 1.4Vpp.
- La salida del conversor CXA no puede cargar con 150Ω en acoplamiento directo, pero si puede cargar con 150Ω desacoplando la componente continua.

Los valores para los componentes son los que se muestran en la figura, es decir, R24 y R25 = 120Ω, R26 = 150Ω, C34 = 220uF y C35 = 22uF.

El valor de la capacidad de entrada (C35) es el recomendado por el fabricante, y se supone que ofrece una impedancia despreciable a la frecuencia de trabajo. El condensador C34 bloquea la componente continua de salida del conversor de vídeo CXA1645, evitando de esta forma la limitación de salida antes mencionada y evitando también un posible cortocircuito al accionar el interruptor de supresión.

Con todo lo expuesto los resultados obtenidos son los siguientes:

- **Zin** ≈ 60Ω // 150Ω = 42.8Ω.
- **Vin** ≈ (1.4V * 150Ω) / 210Ω = 1V.
- **Io** ≈ 1.4V / 210Ω = 6.6mA.

Tal como se puede observar en los cálculos se cumplen todos los requerimientos de partida.

- **Adaptación de entrada SYNC:** La adaptación de entrada para la señal de sincronismo compuesto que proporciona el microcontrolador es idéntica a la utilizada en el conversor anterior, por lo que no es necesaria su explicación.

Conexiones de salida: En este caso tenemos un único circuito de salida. Su diagrama se muestra en la figura 4.32, y su función es la de ajustar la impedancia y el nivel en la salida de vídeo compuesto.

Como la salida del amplificador produce 2Vpp sobre una carga de 150Ω, con una resistencia de salida de 75Ω se tendría una tensión de 1Vpp sobre una carga estándar de 75 Ω.

Pero en algunos casos la entrada del monitor tiene un gran condensador de desacoplo, por lo que es necesario conectar una resistencia de 150 Ω entre el terminal de

salida y tierra, para de esta forma proveer de un camino de baja impedancia para descargar el condensador, teniendo esta solución como inconveniente un notable incremento de la corriente de salida (unos 30mA).

Teniendo en cuenta todo lo expuesto se adopta la siguiente solución:

- Se conecta una resistencia variable de 150Ω entre el terminal de salida y tierra, para que de esta forma no existan problemas en los monitores con desacoplo de la señal de entrada. Además, la resistencia variable permitirá el control del nivel de salida.
- Se conecta un condensador de 220uF en serie con la línea de salida, para de esta forma evitar una corriente excesiva en caso de monitores con acoplamiento directo de la señal de entrada.
- Se conecta una carga de 75Ω en serie con la línea de salida, que aproxima la impedancia de salida a su valor estándar.

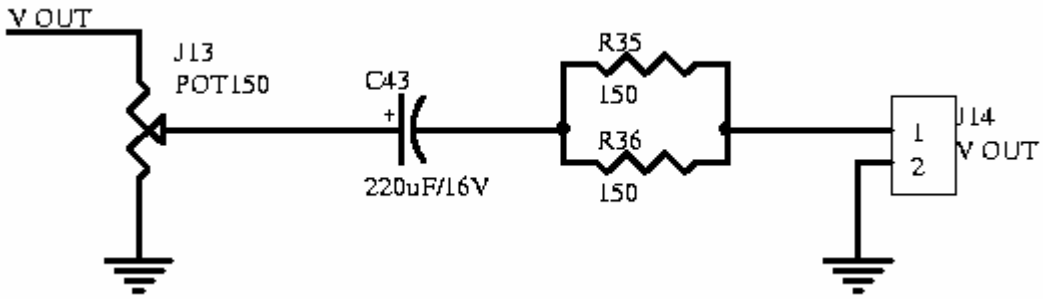


Figura 4.32 - Circuito de salida en el conversor MC1377.

4.2. DESCRIPCIÓN Y FUNCIONAMIENTO DEL MONITOR.

DESCRIPCIÓN Y FUNCIONAMIENTO DEL MONITOR

Cuando queremos visualizar líneas de vídeo en un osciloscopio nos encontramos con una serie de inconvenientes:

1) No podemos distinguir el campo y número de línea de la señal en pantalla.
2) El desplazamiento a través de las líneas solo es posible si se utiliza un retardo variable después del disparo o un osciloscopio digital, y en ninguno de los casos será posible realizar un gran desplazamiento.
3) No es posible la separación de las señales en función de su utilidad (vídeo, teletexto y control de calidad).

Por todos los motivos expuestos, se vio la necesidad de incluir en el equipo didáctico un Monitor de Vídeo capaz de solventar todos los problemas existentes.

El monitor de vídeo podría funcionar de la siguiente forma:

1) Detectaría el instante en que comienza cada campo.
2) Una vez comienza el campo deseado contaría las líneas de vídeo.
3) Cuando se alcance la línea deseada dispararía el osciloscopio para poder visualizarla.

Pero el monitor de vídeo no solo tiene que realizar las funciones antes mencionadas, sino que además:

1) Debe aceptar las instrucciones provenientes de un teclado que permitirá gobernar el equipo.
2) Debe mostrar en un display la información correspondiente al modo de funcionamiento, el campo y número de línea.
3) Tiene que proporcionar una alerta sonora que indique los cambios importantes en el equipo.

Lo primero que debe definirse es la cantidad y tipo de modos de funcionamiento que tendrá el equipo, los controles que habrá de poseer, y las especificaciones del mismo, ya que esto determinará las características, y por tanto la complejidad, del sistema a desarrollar.

Especificaciones del sistema.

El equipo tendrá cuatro modos de funcionamiento:

1) Modo Vídeo (Solo visualiza líneas de vídeo).
2) Modo T.X.T. (Solo visualiza líneas de teletexto).
3) Modo V.I.T. (Solo visualiza líneas de control de calidad).
4) Modo Continuo (Se visualizan todas las líneas).

Se requieren controles para seleccionar el modo de funcionamiento, el campo de la señal, y el número de línea. De forma adicional se podrá incluir un control para la velocidad de cambio de las líneas (en el caso de que sean muchas).

Estos controles se realizaran a través de un teclado matricial situado a lo largo del display. Cada tecla se situará a la altura de una de las líneas del display, de manera que cada tecla sirva para seleccionar la opción del menú que se muestra en la línea. De esta forma cada tecla ejecutará una acción diferente dependiendo del menú que se muestre en pantalla.

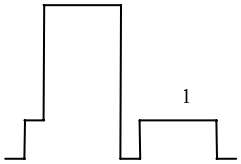
La señal de entrada deberá ser de 1Vpp sobre una carga de 75Ω.

Por ultimo, si es posible se utilizará como buzzer un simple altavoz.

Cuadro de imagen en TV.

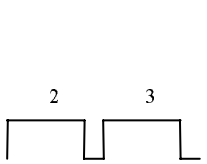
A continuación, partiendo de una imagen caracterizada por tener todas sus líneas de imagen completamente blancas, se describe detalladamente lo que representa cada línea de la señal de televisión.

Líneas 623,624,625:



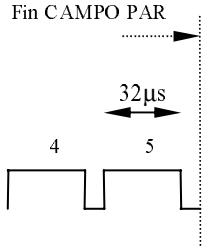
Línea 623

La primera mitad se corresponde con la última (media) línea de imagen del campo par. La segunda mitad es el primer impulso de igualación anterior.



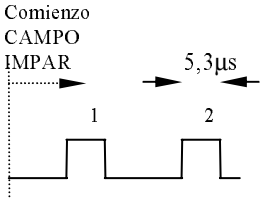
Línea 624

Al primer impulso de igualación anterior le siguen otros cuatro impulsos de igualación. En total son 5 impulsos de igualación anteriores cuya duración en conjunto es de 2,5 veces el tiempo total de línea.



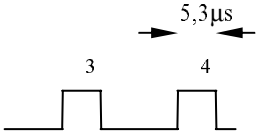
Línea 625

Líneas 001,002,003:

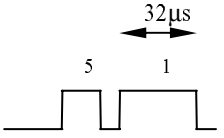


Línea 001

Ahora se comienzan a enviar los impulsos de sincronismo vertical. En la línea 001 se envían el primero y el segundo y, en la línea 002 el tercero y el cuarto.



Línea 002



Línea 003

La primera parte es el quinto impulso de sincronismo vertical. La segunda parte es el primer impulso de igualación posterior.

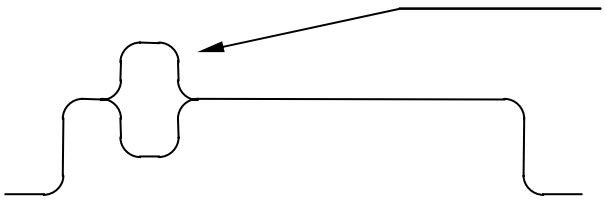
El conjunto de cinco impulsos de sincronismo vertical tienen una duración total de 2,5 veces el tiempo total de línea.

Líneas 004,005:

Línea 004 Línea 005

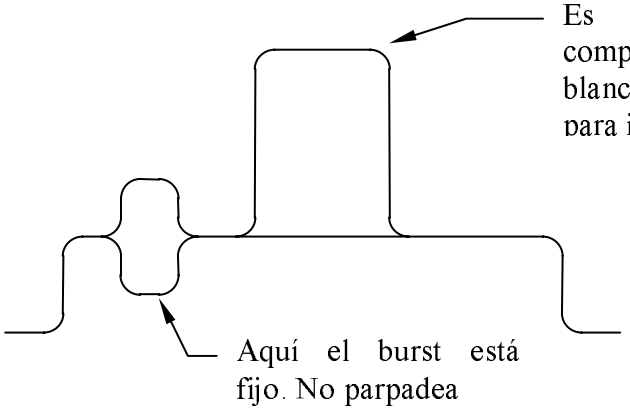
Al primer impulso de igualación posterior (línea 003) le siguen otros cuatro impulsos de igualación posterior. En total son 5 impulsos de igualación posteriores cuya duración en conjunto es de 2,5 veces el tiempo total de línea.

Línea 006:



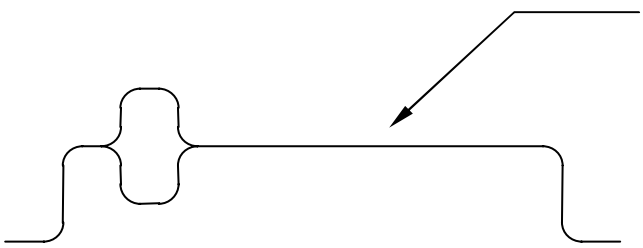
El burst aparece parpadeando. Es decir, aparece en una imagen, y en la siguiente no aparece... así sucesivamente...
Si en esta línea al burst le correspondiera una fase de +135°, entonces sí aparece.
Si por el contrario le tocara llevar una fase de -135°, entonces no aparecería.

Línea 007:



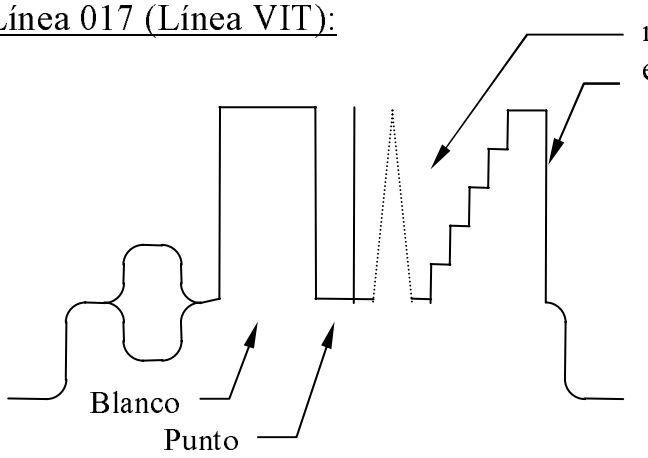
Es un blanco intermitente. Existe para compatibilizar distintos sistemas PAL. Este blanco aparece una vez cada 8 campos. Sirve para indicar cuándo comienza la secuencia.

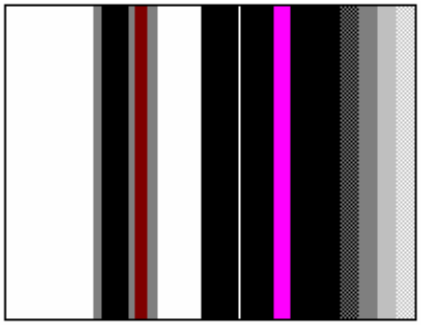
Líneas [008,016]:



En estas líneas es posible introducir información de teletexto. En la práctica, el generador de cartas patrón no está insertando esta información de teletexto.

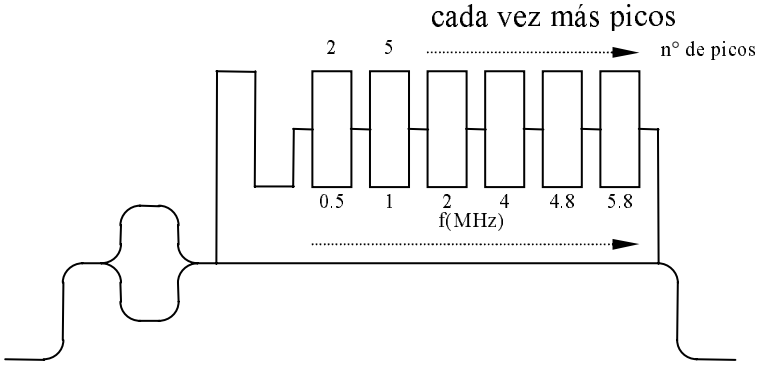
Línea 017 (Línea VIT):



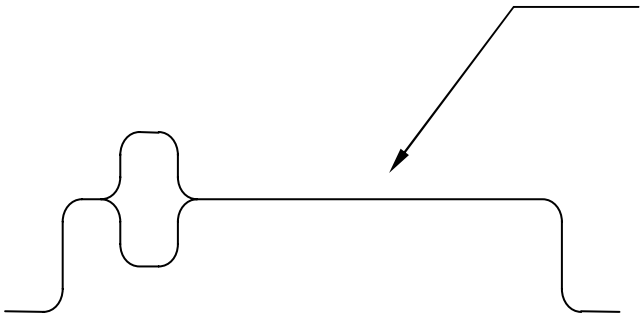


Las líneas VIT son señales de prueba que se insertan en las líneas 17,18,330 y 331 para poder realizar medidas de calidad del sistema de telecomunicación.

Línea 018 (Línea VIT):

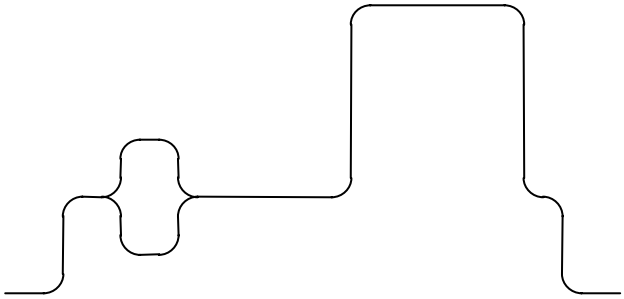


Líneas 019,020,021,022:



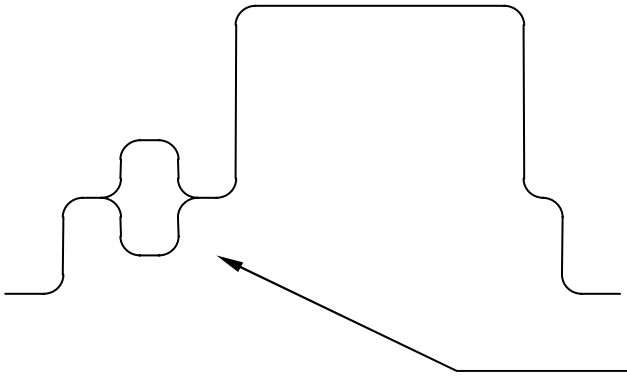
En estas líneas es posible introducir información de teletexto. En la práctica, el generador de cartas patrón no está insertando esta información de teletexto. Estas líneas son semejantes a las líneas: de 008 a 016 ambas inclusive.

Línea 023:



La segunda mitad de la línea 023 se corresponde con media línea de imagen (primera línea de imagen).

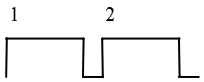
Líneas [024,310]:



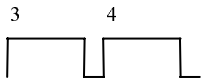
La información de imagen que llevan asociada es la misma para todas estas líneas.

Burst parpadeante en la línea 310

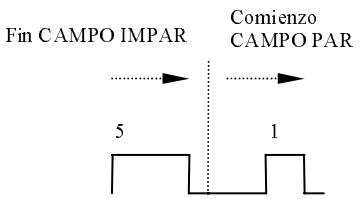
Líneas 311,312,313:



Línea 311



Línea 312

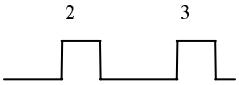


Línea 313

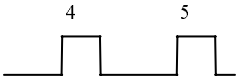
Se comienzan a enviar los cinco impulsos de igualación anteriores del campo impar. Pero en estas dos líneas sólo se envían los cuatro primeros. En total son 5 impulsos de igualación anteriores cuya duración en conjunto es de 2,5 veces el tiempo total de línea.

La primera mitad se corresponde con el quinto impulso de igualación anterior. En total son 5 impulsos de igualación anteriores cuya duración en conjunto es de 2,5 veces el tiempo total de línea. La segunda mitad se corresponde con el primer impulso de sincronismo vertical.

Líneas 314,315:



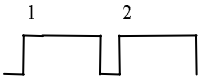
Línea 314



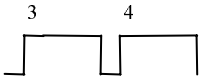
Línea 315

Se terminan de enviar los cinco impulsos de sincronismo vertical. En total, estos 5 impulsos de sincronismo vertical tienen una duración de 2,5 veces el tiempo total de línea.

Líneas 316,317:



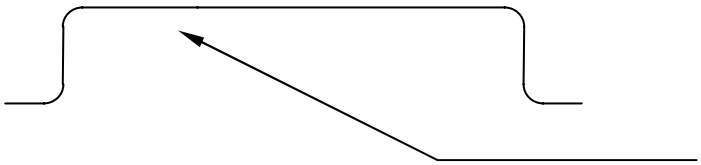
Línea 316



Línea 317

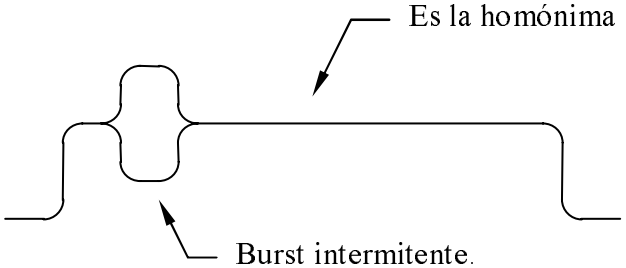
Estos cuatro impulsos son los cuatro impulsos de igualación posterior del campo par. La duración total de los impulsos de igualación anterior es de 2 veces el tiempo de una línea.

Líneas 318:

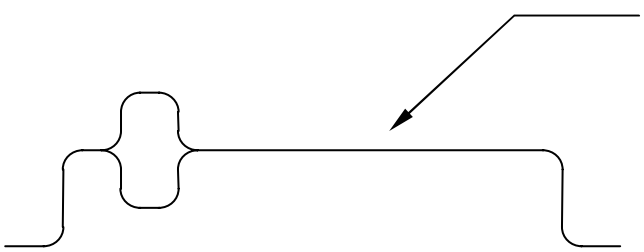


Es la última línea sin burst (si le tocaba una burst de -135°) o la penúltima sin burst (si le tocaba una burst de +135°)

Líneas 319:

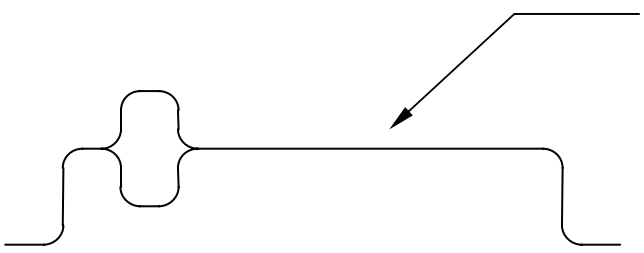


Línea 320:



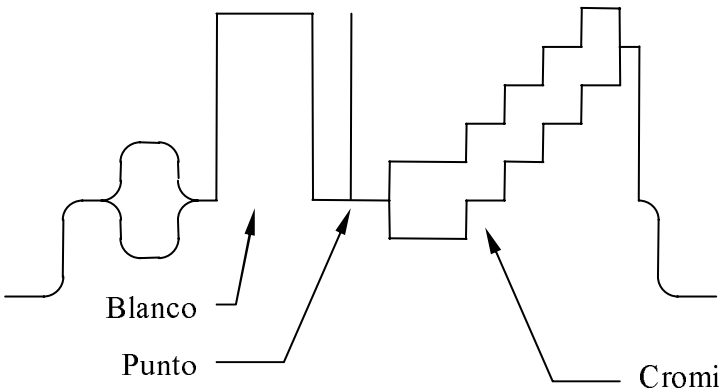
Es la homónima de la línea 007 en el campo par. Pero no existe blanco intermitente.

Líneas [321,329]:



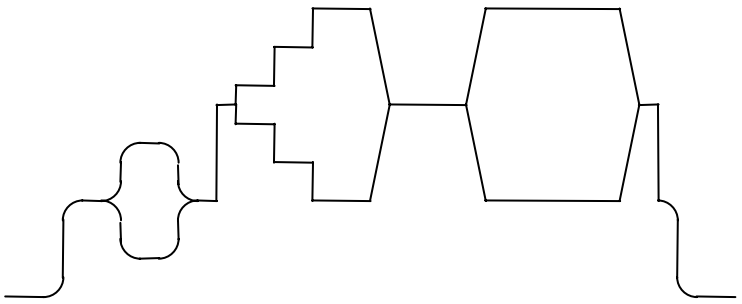
En estas líneas es posible introducir información de teletexto. En la práctica, el generador de cartas patrón no está insertando esta información de teletexto.

Líneas 330 (Línea VIT):



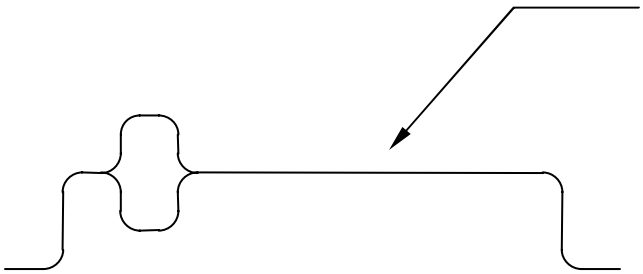
Crominancia constante (en cada escalón hay los mismos picos)
Luminancia diferente

Líneas 331 (Línea VIT):



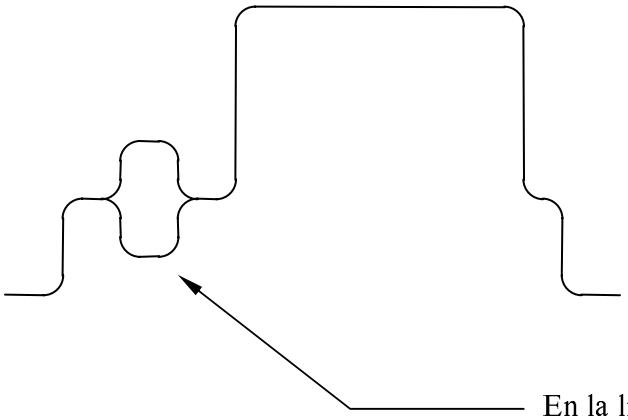
Un mismo color a diferentes valores de saturación: 20%, 60%, 100%

Líneas [332,335]:



En estas líneas es posible introducir información de teletexto. En la práctica, el generador de cartas patrón no está insertando esta información de teletexto.

Líneas [336,622]:



La información de imagen que llevan asociada es la misma para todas estas líneas.

En la línea 622 aparece el burst intermitente.

Solución para el diseño del monitor.

Para cumplir las especificaciones se decidió incluir los siguientes bloques:

1) **Separador de Sincronismos Activo LM1881N.**

Procesa la señal de vídeo y extrae la información de sincronismos en las que se basa el funcionamiento del monitor.

Recibe por su entrada la señal de vídeo compuesto, y a continuación extrae la información de sincronismo de la señal suministrando las siguientes salidas:

- Salida de sincronismo compuesto: Contiene toda la información de sincronismo, es decir, tanto sincronismo vertical como sincronismo horizontal.
- Salida de sincronismo vertical: Suministra un nivel bajo durante aproximadamente 230us a partir del primer impulso de sincronismo horizontal del campo (ish de la primera línea).
- Salida de campo: Suministra un nivel alto durante el campo impar, y un nivel bajo durante el campo par.

2) **Microcontrolador PIC16F876-04**.

Procesa la información que recibe del teclado y del separador de sincronismos activo, y genera la información necesaria para el control del display, el buzzer y el disparo del osciloscopio.

Para cumplir estas tareas hace lo siguiente:

- Utiliza los Bits 0, 1, y 2 del PORTA, configurados como salidas, para el bus de control del display.
- Utiliza los Bits 0 a 7 del PORTB, configurados como salidas, para el bus de datos del display.
- A través del Bit 3 del PORTA, configurado como entrada, obtiene la información de campo proveniente del separador de sincronismos.
- A través del Bit 4 del PORTA, configurado como entrada de reloj para el contador TMR0, obtiene la información de sincronismo vertical proveniente del separador de sincronismos. El contador se carga con su valor máximo, para que con cada impulso que indica el comienzo del campo se desborde y produzca una interrupción. Esta interrupción prepara el contador TMR0 para una nueva interrupción y carga el contador TMR1 con el número de líneas a contar.
- A través del Bit 0 del PORTC, configurado como entrada de reloj para el contador TMR1, obtiene la información de sincronismo compuesto suministrada por el separador de sincronismos. Este contador ascendente de 16 Bits lleva la cuenta de todos los impulsos de sincronismo. Se carga con el complemento a dos del número de impulsos que se quieren contar, y cuando se desborda provoca una interrupción. La interrupción ha de provocar el disparo del osciloscopio si el campo de la señal es el adecuado.
- Utiliza el Bit 1 del PORTC, configurado como salida, para el control de un conmutador analógico (deja pasar la señal en el instante adecuado).
- Utiliza el Bit 2 del PORTC, configurado como salida, para el control del buzzer.

- Por ultimo, utiliza el PORTB para el control del teclado matricial, para lo cual configura los Bits 0 a 3 como salidas y los Bits 4 a 7 como entradas. El control del teclado se realiza mediante interrupciones cada vez que hay un cambio en el PORTB.

3) **Conmutador Analógico CD4066BP.**

El conmutador analógico recibe la señal de vídeo por una de sus entradas, y la deja pasar cuando el microcontrolador se lo indica. El osciloscopio se conectará a la carga de salida del conmutador , y se disparará justo cuando aparezca la señal.

Aquí en el Monitor de Vídeo la circuiteria externa es mínima, ya que casi todo el procesamiento es digital.

A continuación, se comentaran cada uno de los bloques, describiendo detalladamente las características de cada uno y las conexiones de estos con el resto del circuito.

Dentro de la descripción del microcontrolador también se incluirá la descripción del teclado, el display y el buzzer, ya que todos ellos interactúan solo con el microcontrolador.

El software se comentará más adelante en el capitulo 5.

4.2.1. MICROCONTROLADOR PIC16F876.

4.2.1.1. Características técnicas.

Se trata de un microcontrolador de 8 bits, con memoria de tipo Flash/EEPROM, y cápsula de 28 terminales.

Las principales características proporcionadas por el fabricante son las siguientes:

Características de la CPU:

- CPU tipo RISC de altas prestaciones.
- Solo 35 instrucciones.
- Todas las instrucciones se ejecutan en un solo ciclo, excepto las instrucciones de salto, que requieren dos ciclos.
- Velocidad de operación:
  - Entrada de reloj -> DC – 20MHz.
  - Ciclo de instrucción -> DC – 200ns.
- 8K x 14 posiciones de memoria de programa tipo Flash.
- 368 Bytes de memoria de datos tipo RAM.
- 256 Bytes de memoria de datos tipo EEPROM.
- Compatible Pin a Pin con los PIC16C73B/74B/76/77.
- Capacidad de Interrupción (hasta 14 posibles fuentes).
- Pila hardware de 8 niveles.
- Modos de direccionamiento directo, indirecto, y relativo.
- Power-on Reset (POR).
- Power-up Timer (PWRT) y Oscillator Start-up Timer (OST).
- Perro guardián.
- Protección del código.
- Modo reposo de bajo consumo.
- Oscilador seleccionable.
- Tecnología CMOS FLASH/EEPROM de alta velocidad y bajo consumo.

- Diseño totalmente estático.
- Programación serie en circuito a través de dos terminales (ICSP).
- Capacidad de programación serie en circuito con alimentación simple de 5V.
- Depuración en circuito a través de dos terminales.
- Gran margen del voltaje de operación: 2V a 5.5V.
- Alta corriente de entrada/salida: 25mA.
- Bajo consumo:
  - < 0.6mA para 4V y 4MHz.
  - 20 uA para 3V y 32KHz.
  - < 1uA en estado de espera.

Características de los Periféricos:

- Temporizador/Contador TMR0 de 8 Bits con predivisor de 8 Bits.
- Temporizador/Contador TMR1 de 16 Bits con predivisor, puede ser incrementado durante el modo de reposo a través de un oscilador externo.
- Temporizador/Contador TMR2 de 8 Bits, con registro de periodo de 8 Bits, predivisor, y posdivisor.
- Dos módulos de captura, comparación y PWM.
  - Captura de 16 Bits, con resolución máxima de 12.5ns.
  - Comparación de 16 Bits, con resolución máxima de 200ns.
  - PWM con resolución máxima de 10 Bits.
- Conversor analógico-digital multi-canal de 10 Bits.
- Puerto serie sincrono (SSP) con SPI (Modo Maestro) y I2C (Maestro/Esclavo).
- Transmisor/Receptor sincrono/asíncrono universal (USART/SCI), con detección de dirección de 9 Bits.
- Puerto Paralelo esclavo (PSP) de 8 Bits, con controles externos /RD, /WR y /CS (Solo modelos con 40/44 patillas).
- Circuito de detección de fallo en la alimentación para reset por fallo en alimentación (Brown-out Reset, BOR).

Características eléctricas:

- Temperatura ambiente bajo polarización.....-55°C a +125°C
- Temperatura de almacenaje.....-65°C a +150°C
- Voltaje en algún pin con respecto a VSS (excepto VDD, /MCLR y RA4)..... -0.3V a (VDD + 0.3V)
- Voltaje en VDD con respecto a VSS.....-0.3 a +7.5V
- Voltaje en /MCLR con respecto a VSS.....0 a +14V
- Voltaje en RA4 con respecto a VSS.....0 a +8.5V
- Disipación de potencia total.....1W
- Máxima corriente de salida por VSS.....300 mA
- Máxima corriente de entrada por VDD.....250 mA
- Corriente de entrada por borne, IIK (VI < 0 o VI > VDD).....± 20 mA
- Corriente de salida por borne, IOK (VO < 0 o VO > VDD) .....± 20 mA
- Máxima corriente de entrada por algún terminal de I/O..... 25 mA
- Máxima corriente de salida por algún terminal de I/O.....25 mA
- Máxima corriente de entrada por alguna Puerta .....200 mA
- Máxima corriente de salida por alguna Puerta.....200 mA

Tabla 4.4 - Descripción de terminales del microcontrolador PIC16F876.

Nombre	Número	Tipo I/O/P	Tipo Buffer	Descripción
OSC1/CLKIN	9	I	ST/CMOS	Entrada oscilador cristal/ entrada reloj externo.
OSC2/CLKOUT	10	O	-	Salida oscilador cristal/ salida ¼ reloj externo.
/MCLR/VPP	1	I/P	ST	Reset principal. Entrada voltaje programación.

RA0 RA1 RA2 RA3 RA4/T0CKI RA5	2 3 4 5 6 7	I/O I/O I/O I/O I/O I/O	TTL TTL TTL TTL ST TTL	Puerto de entrada/salida bidireccional. RA4 también puede ser entrada de reloj del TMR0.
RB0/INT RB1 RB2 RB3 RB4 RB5 RB6 RB7	21 22 23 24 25 26 27 28	I/O I/O I/O I/O I/O I/O I/O I/O	TTL/ST TTL TTL TTL TTL TTL TTL/ST TTL/ST	Puerto de entrada/salida bidireccional. Resistencias de pull-up. RB0 capacidad de INT. RB4-7 capacidad de INT. por cambio de estado. RB6-RB7 reloj-datos programación serie.
RC0/T1CKI RC1 RC2 RC3 RC4 RC5 RC6 RC7	11 12 13 14 15 16 16 18	I/O I/O I/O I/O I/O I/O I/O I/O	ST ST ST ST ST ST ST ST	Puerto de entrada/salida bidireccional. RC0 también puede ser entrada de reloj del TMR1.
VSS	8-19	P	-	Tierra
VDD	20	P	-	Alimentación positiva

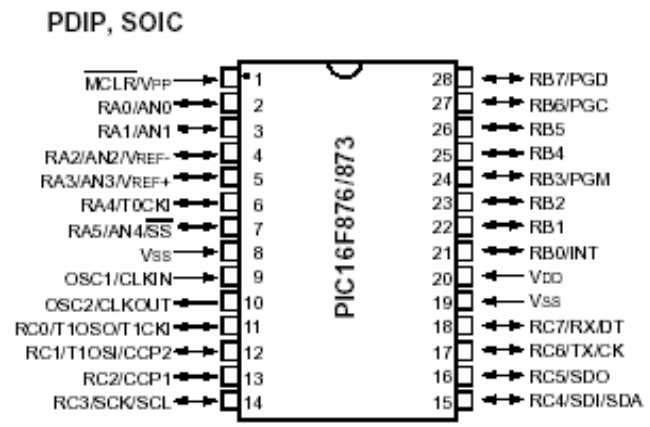


Figura 4.33 - Diagrama de terminales del microcontrolador PIC16F876.

4.2.1.2. Temporizadores/Contadores.

Una exigencia en las aplicaciones de control es la regulación estricta de los tiempos que duran las diversas acciones que realiza el sistema. El dispositivo típico destinado a gobernar los tiempos recibe el nombre de temporizador o "timer" y, básicamente, consiste en un contador ascendente o descendente que determina un tiempo determinado entre el valor que se le carga y el momento en que se produce su desbordamiento o paso por 0.

En la gama baja los microcontroladores PIC sólo disponen de dos temporizadores. Uno de ellos actúa como Principal y sobre él recae el control de tiempos de las operaciones del sistema. El otro recibe el nombre de Perro guardián o "Watchdog". El Perro guardián vigila que el programa no se "cuelgue" y dejen de ejecutarse las instrucciones secuenciales del mismo tal como lo ha previsto el diseñador. Para realizar esta labor de vigilancia, el Perro guardián da un paseo por la UCP cada cierto tiempo y comprueba si el programa se ejecuta normalmente; en caso contrario, por ejemplo si el control está detenido en un bucle infinito o a la espera de algún acontecimiento que no se produce, el perro ladra y provoca un reset, reinicializando todo el sistema.

Tanto el Temporizador principal, TMR0, como el Perro guardián, WDT, a veces precisan controlar tiempos largos y aumentar la duración de los impulsos de reloj que les incrementan o decrementan. Para cubrir esta necesidad, se dispone de un circuito programable llamado Divisor de frecuencia que divide la frecuencia utilizada por diversos rangos para poder realizar temporizaciones más largas.

Para regular el comportamiento del Temporizador principal, el Perro guardián y el Divisor de frecuencia se emplean algunos bits de la Palabra de configuración y de1 Registro de opciones (OPTION).

El Divisor de frecuencia puede aplicarse a uno de los dos temporizadores, al TMR0 o al WDT. Con el Temporizador principal actúa en primer lugar, o sea, los impulsos pasan primero por el Divisor de frecuencia y, una vez aumentada la duración de los mismos, se aplican a TMR0. Actúa como Divisor previo o "Prescaler". Con el Perro guardián, el Divisor de frecuencia actúa después ("Post-scaler").

El Divisor de frecuencia puede actuar al ritmo de una señal externa aplicada sobre la tita T0CKI, o bien, con la señal de reloj interna del microcontrolador CLKOUT, procedente del oscilador propio. Mediante algunos bits del Registro de opciones y la Palabra de configuración se controla el trabajo del Divisor de frecuencia sobre el TMR0 o el WDT.

Temporizador/Contador TMR0.

Se trata de un contador ascendente de 8 bits que puede actuar de dos formas :

1) Se le introducen los impulsos desde el exterior por la patita T0CKI. Su misión es "contar" el número de acontecimientos externos.
2) Trabaja y cuenta los impulsos de reloj del oscilador interno (CLKOUT). Se usa para determinar un tiempo fijo. Estos impulsos tienen una duración conocida que es la de un ciclo de instrucción cuya frecuencia es la cuarta parte del oscilador principal (Fosd4).

El TMR0 se comporta como un registro de propósito especial ubicado en la posición 1 del área de datos. Puede ser leído y escrito al estar conectado directamente al bus de datos. Como se trata de un contador ascendente, conviene cargarle con el valor de los impulsos que se desean contar pero en forma de complemento a 2. Así, si se quieren contar cuatro impulsos de reloj se carga al TMR0 con el complemento a 2 de 4, lo que significa cargarle con -4. De esta manera, con la llegada de cuatro impulsos se alcanza valor 0, que determina el tiempo a controlar.

Para conocer el estado en que va la cuenta del TMR0 se el puede leer en cualquier momento. Cuando se escribe un nuevo valor sobre TMR0 para iniciar una nueva temporización, el incremento del mismo se retrasa durante los dos ciclos de reloj posteriores.

Temporizador/Contador TMR1.

Algunos modelos de la gama media además de disponer del temporizador TMR0 poseen otros dos llamados TMR1 y TMR2.

El TMR1 se trata de un Temporizador/Contador ascendente de 16 bits, por lo que está implementado mediante dos registros específicos TMRlH y TMRlL, que contienen el valor del contaje en cada momento. El valor del registro TMRlH-TMRlL evoluciona desde 0000 h hasta FFFF h, en cuyo instante activa el señalizador TMR1IF y vuelve a 0000 h. Como fuente de los impulsos de reloj existen tres alternativas :

1) Generación interna (4 Tosc).
2) Generación mediante un oscilador externo controlado por cristal que se conecta a las patitas RC0/T1OSO/TICKI y RC1/T1OSI/CCP2. El oscilador se activa poniendo a 1 el bit T1OSCEN del registro T1CON.

   El bit TMR1CS del registro T1CON selecciona entre reloj interno o externo.
3) Trabaja en modo contador de eventos, cuando los impulsos externos a contar se aplican a la patita RC0/T1OSO/TlCKI.

La fuente de los impulsos de reloj se aplica a un Divisor de Frecuencia (Prescaler) que los divide por 1, 2, 4 ú 8, según el valor de los bits <1:0> (TlCKPS) del registro TlCON. El reloj externo puede estar sincronizado o no con el interno, según el bit TlSYNC de TlCON. El interno siempre es síncrono. El periodo en TlCKI es preciso que tenga una duración mínima de 4.Tosc.

En el modo de Reposo cuando funciona en modo síncrono, el TMR1 deja de incrementarse pues se desconecta el circuito de sincronismo. En forma asíncrona el TMRI sigue contando durante el modo de reposo, por eso se puede emplear como reloj de tiempo real y para sacar del modo de Reposo al sistema.

También en modo asíncrono se puede usar como base de tiempos en operaciones de Captura y Comparación. El módulo CCP pone a 0 el TMR1 cuando se produce una Captura o una Comparación.

Registro T1CON.

A continuación se presenta la misión de los bits de este registro que ocupa la posición 10 h del banco 1.

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	T1CKPS1	T1CKPS0	T1OSCEN	/T1SYNC	TMR1CS	TMR1ON

TMR1ON: Activación del TMR1.

1 = TMR1 activado.

0 = TMR1 desactivado.

TMR1CS: Selección de reloj para el timer 1.

1 = Externo por el pin RC0/TCKI (flanco ascendente).

0 = Reloj interno (4Tosc).

/T1SYNC: Control de sincronismo del reloj externo.

1 = Reloj externo no sincronizado.

0 = Reloj externo sincronizado.

T1OSCEN: Activación del oscilador por cristal para el TMR1.

1 = Oscilador activado.

0 = Oscilador desactivado.

T1CKPS <1:0>: Selección del preescaler para el TMR1.

11 = 8.

10 = 4.

01 = 2.

00 = 1.

4.2.1.2.1. Modo de operación de los contadores.

A continuación se describen las tareas encomendadas a los contadores TMR0 y TMR1, así como los modos de operación y configuración de los mismos.

Contador TMR0.

Se puede decir que en este caso el TMR0 no se utiliza para contar o temporizar, sino que se hace uso de el por su capacidad para provocar interrupciones. La función que se le asigna, es la de detectar el instante en que comienza el barrido de los campos, para de esta forma comenzar a contar las líneas de vídeo tal como se explico en la descripción del monitor.

La entrada de reloj externa del contador (RA4/T0CKI) se conecta con la salida de sincronismo vertical que suministra el separador de sincronismos activo LM1881N. El contador (ascendente) se encuentra cargado con su valor máximo (255), y se configura para que su entrada sea sensible al flanco descendente de la señal. Para seleccionar la entrada de reloj externa se activa el Bit T0CS, y para seleccionar sensibilidad al flanco descendente se activa el Bit T0SE, ambos del Registro de Opciones.

Cuando comienza el barrido de un nuevo campo, exactamente con la primera línea, el separador de sincronismos suministra un nivel bajo por su salida de sincronismo vertical, que dura aproximadamente 230uS. Con el flanco descendente de esta señal aumenta el contenido del contador TMR0, y este se desborda y provoca una interrupción. Para habilitar la interrupción se ha de activar el Bit T0IE del Registro de Interrupciones.

De esta forma se detecta el comienzo de los campos. Seguidamente vemos que se hace después de la detección.

La rutina de tratamiento de interrupción para el TMR0 realiza las siguientes tareas:

1) Comprueba el modo de funcionamiento.

2) Si está en el menú principal, dispara, y se visualizan parte de los impulsos de sincronismo vertical. Pero si por el contrario se encuentra dentro de alguno de los modos de funcionamiento, testea el campo leyendo el Bit 3 del PORTA.
3) Si está en alguno de los modos de funcionamiento, y además el campo es el correcto, entonces carga el contador TMR1 con el valor adecuado para que se produzca el disparo en la línea deseada.
4) Por ultimo, se actualiza el contador TMR0 para que se vuelva a producir la interrupción con el comienzo del próximo campo.

En la figura 4.34 se muestra la sección código encargada del control del TMR0. El código se ha escrito en lenguaje C para una mayor comodidad, ya que no se requieren temporizaciones complejas como en el caso del generador.

```
//RSI Desbordamiento de tmr0
void tmr0_int(void)
{
 if(modo != 0)                    //si no está en menú principal
 {
  field = input_pin_port_a(3);    //determina campo en la señal de video
  if(field == campo)              //si está en campo seleccionado
  {
   tmr1l = tmr1b;                 //actualiza tmr1l
   tmr1h = tmr1a;                 //actualiza tmr1h
  }
 }
 else                             //si está en menú principal
 {
  set_bit(portc,1);               //dispara
  clear_bit(portc,1);
 }
 tmr0 = 0xFF;              //actualiza tmr0 para interrupción por ISV
}
```

Figura 4.34 – Software de la RSI asociada a TMR0.

Contador TMR1.

El contador TMR1 si se utiliza como tal, es decir, tiene asignada la función de contar sucesos. La función exacta del TMR1 es la de contar las líneas de vídeo, para de esta forma saber cuando se ha de producir el disparo.

La entrada de reloj externa (RC0/T1CKI) del contador se conecta con la salida de sincronismo compuesto que proporciona el separador de sincronismos, para de esta forma poder contar todos los impulsos de interés (isv, iip e ish). El contador (ascendente) se encuentra cargado con el numero de líneas a contar en complemento a dos, es decir, con el número de líneas que se han de contar para que el contador se desborde y produzca una interrupción. Se ha de habilitar el contador activando el Bit TMR1ON del registro T1CON. Para seleccionar la entrada de reloj externa se ha de activar el Bit TMR1CS, y para que los impulsos de reloj no estén sincronizados con el reloj interno se ha de activar el Bit /T1SYNC, ambos del registro T1CON.

Conociendo el número exacto de impulsos (isv, iip e ish) antes de la línea que se pretende visualizar, se puede cargar el contador TMR1 para que se desborde justo antes de la línea deseada y provoque una interrupción. Para habilitar la interrupción se ha de activar el Bit PEIE del Registro de Interrupciones, junto con el Bit TMR1IE del Registro PIE1.

De esta forma se detecta la llegada de la línea. Seguidamente la rutina de tratamiento de interrupción para el TMR1, mantiene el conmutador analógico en conducción el tiempo suficiente para que se vea en el osciloscopio la línea seleccionada. El código asociado a esta interrupción se muestra a continuación.

```
//RSI Desbordamiento de tmr1
void tmr1_int(void)
{
 set_bit(portc,1);              //dispara,
 delay_us(115);                 //y retarda
 clear_bit(portc,1);            //para visualizar línea
}
```

Figura 4.35 – Software de la RSI asociada a TMR1.

4.2.1.3. Interrupciones.

Una interrupción consiste en una detención del programa en curso para realizar una determinada rutina que atienda la causas que ha provocado la interrupción. Es como una llamada a subrutina, que se origina por una causa diferente a la de una instrucción del tipo CALL. Tras la terminación de la rutina de interrupción, se retorna al programa principal en el punto en que se abandonó.

Las causas que originan una interrupción pueden ser externas, como la activación de una patita con el nivel lógico apropiado, e internas, como las que pueden producirse al desbordarse un temporizador como el TMR0.

En las aplicaciones industriales, las interrupciones son un recurso muy potente para atender los acontecimientos físicos en tiempo real. Las interrupciones evitan que la UCP explore continuamente el nivel lógico de una patita o el valor de un contador.

En nuestro caso nos servimos de tres posibles fuentes de interrupción:

- Desbordamiento del TMR0.
- Desbordamiento del TMR1.
- Cambio de estado en las líneas de la Puerta B.

Cada causa de interrupción está controlada mediante dos líneas o señales. Una de ellas actúa como un señalizador que indica si se ha producido o no el acontecimiento, mientras que la otra es el permiso o prohibición de la interrupción en sí. Así, por ejemplo, en la figura 4.36 se muestra el esquema de control de la generación de la interrupción por desbordamiento del TMR0. Para que el desbordamiento de TMR0 provoque una interrupción es necesario que las dos señales de entrada de la puerta AND tengan nivel alto. TOIF es el señalizador que indica si se ha producido el rebosamiento y TOIE es el bit de permiso de la interrupción.

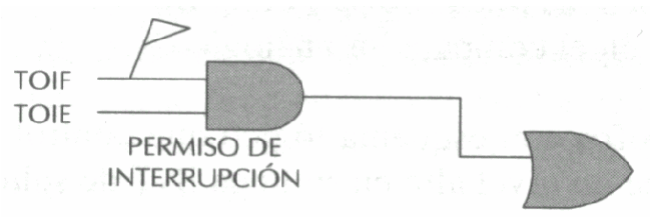


Figura 4.36 – Circuito de control para la interrupción del TMR0.

El valor que se aplica a las señales de entrada del circuito de gobierno de las interrupciones proviene del que tengan los bits de los registros INTCON, PIR1 y PIEl.

El bit GIE de activación global del permiso de interrupción, situado en el registro INTCON, se borra automáticamente cuando se reconoce una interrupción para evitar que se produzca otra cuando se atiende a la inicial. Al retornar de la interrupción, el bit GIE se vuelve a activar.

Como la interrupción puede producirse por diversas causas, el software inicial de 1a rutina de interrupción comienza explorando cuál ha sido la causa que la ha provocado.

Los señalizadores de las fuentes de interrupción se deben borrar por software, después de atender a una de ellas, para prevenir falsas interrupciones.

La interrupción externa, producida por la activación de una patita, debe mantenerse con el nivel activo al menos 3 ó 4 ciclos de instrucción, dependiendo de cuándo se produce.

Fases de una interrupción.

Se expone ordenadamente la secuencia de acciones que se llevan a cabo cuando se atiende a una interrupción.

1) Se activa una posible causa de interrupción. El señalizador de dicha causa, el bit de permiso correspondiente y el global para todas las interrupciones (GIE) están a nivel alto.
2) Para evitar que se produzca otra interrupción mientras dura el tratamiento de la que se ha aceptado, el Bit GIE se pone a 0.
3) El valor actual del PC se guarda en la Pila.
4) El PC se carga con el valor 0004 h, que es el del vector de interrupción.
5) La rutina de interrupción comienza explorando el valor de los señalizadores, para determinar la causa que la ha provocado.
6) Según la causa de la interrupción, la rutina se bifurca a la subrutina correspondiente.
7) Se borran los señalizadores por software, antes de realizar el retorno.

8) Cuando se llega a la última instrucción de la rutina de interrupción, que es la de RETURN, se carga en el PC el valor que inicialmente se guardó en la Pila y se pone el Bit GIE = 1.

En la figura 4.37 se muestra el código de la rutina de tratamiento de interrupción. En ella, se comprueba quien ha provocado la interrupción, se le atiende, y por ultimo se restauran el indicador y los permisos para nuevas interrupciones.

```
//Rutina Servicio Interrupción
void interrupt(void)
{
 clear_bit(intcon,7);          //deshabilita interrupciones
 if(intcon & 4)                //comprueba bandera interrupción tmr0(t0if)
 {
  tmr0_int();                  //RSI desbordamiento de tmr0
  clear_bit(intcon,2);         //borra bandera interrupción tmr0
  set_bit(intcon,5);           //permiso para interrupción tmr0
 }
 if(pir1 & 1)                  //comprueba bandera interrupción tmr1(tmr1if)
 {
  tmr1_int();                  //RSI desbordamiento de tmr1
  clear_bit(pir1,0);           //borra bandera interrupción tmr1
  set_bit(pie1,0);             //permiso para interrupción tmr1
 }
 if(intcon & 1)                //comprueba bandera interrupción portb(rbif)
 {
  portb_int();                 //RSI cambio de estado en puerta b
  clear_bit(intcon,0);         //borra bandera interrupción portb
  set_bit(intcon,3);           //permiso para interrupción portb
 }
 set_bit(intcon,7);            //permiso para interrupciones
}
```

Figura 4.37 – Rutina que da servicio a las interrupciones.

El incremento de las causas de interrupción exige el aumento de los bits de señalización y de autorización, que, según cada modelo, se distribuyen por los registros específicos.

En las aplicaciones industriales es muy frecuente el uso de las interrupciones para el control y atención de los acontecimientos asíncronos exteriores. Por este motivo, al irse elevando por la gama de modelos potentes de los PIC, se aprecia el incremento de las prestaciones que disponen en este aspecto. En la gama baja los modelos carecían de interrupciones, contaban exclusivamente con el Reset. En la gama media hay modelos que soportaban hasta 11 fuentes diferentes de interrupción. Finalmente en la gama alta, versiones como los modernos PICl7C752/756, tienen capacidad para manipular interrupciones multivectoriales.

Para regular las interrupciones en los modelos PIC 16C87X se precisan dos registros, PIR1 y PIR2, que contienen los señalizadores de las causas de interrupción, y otros dos registros, PIE1 y PIE2, que contienen los bits de permiso de las diversas causas. A continuación se muestran la distribución y misión de dichos registros.

Registro PIE1.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE

PSPIE: Bit de habilitación de interrupción por lectura/escritura en el PSP.

1 = Habilitada. 0 = Deshabilitada.

ADIE: Bit de habilitación de interrupción del Conversor A/D.

1 = Habilitada. 0 = Deshabilitada.

RCIE: Bit de habilitación de interrupción por recepción en USART.

1 = Habilitada. 0 = Deshabilitada.

TXIE: Bit de habilitación de interrupción por transmisión en USART.

1 = Habilitada. 0 = Deshabilitada.

SSPIE: Bit de habilitación de interrupción del Puerto serie sincrono.

1 = Habilitada. 0 = Deshabilitada.

CCP1IE: Bit de habilitación de interrupción del CCP1.

1 = Habilitada. 0 = Deshabilitada.

TMR2IE: Bit de habilitación de interrupción del TMR2.

1 = Habilitada. 0 = Deshabilitada.

TMR1IE: Bit de habilitación de interrupción del TMR1.

1 = Habilitada. 0 = Deshabilitada.

Registro PIR1.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE

PSPIE: Bit de habilitación de interrupción por lectura/escritura en el PSP.

1 = Habilitada. 0 = Deshabilitada.

ADIE: Bit de habilitación de interrupción del Conversor A/D.

1 = Habilitada. 0 = Deshabilitada.

RCIE: Bit de habilitación de interrupción por recepción en USART.

1 = Habilitada. 0 = Deshabilitada.

TXIE: Bit de habilitación de interrupción por transmisión en USART.

1 = Habilitada. 0 = Deshabilitada.

SSPIE: Bit de habilitación de interrupción del Puerto serie sincrono.

1 = Habilitada. 0 = Deshabilitada.

CCP1IE: Bit de habilitación de interrupción del CCP1.

1 = Habilitada. 0 = Deshabilitada.

TMR2IE: Bit de habilitación de interrupción del TMR2.

1 = Habilitada. 0 = Deshabilitada.

TMR1IE: Bit de habilitación de interrupción del TMR1.

1 = Habilitada. 0 = Deshabilitada.

4.2.1.4. Puertos de entrada-salida.

El funcionamiento de las puertas de entrada-salida se vio en el apartado correspondiente al Microcontrolador PIC16F84. En este caso solo se mencionaran el número de puertas existentes y la utilidad que se da a cada una de ellas.

El Microcontrolador PIC16F876 dispone de tres puertas denominadas A, B y C, las cuales a su vez disponen de 6, 8 y 8 terminales respectivamente. De todos los periféricos disponibles solo se usan las entradas/salidas digitales y los contadores TMR0 y TMR1, siendo la distribución de terminales la que sigue.

La Puerta A consta de 6 terminales, los cuales se destinan a las siguientes funciones:

- Bits 0 a 2: Estos terminales se encuentran configurados como salidas, las cuales se destinan para el Bus de Control del Display. La distribución de los terminales en orden creciente es: RS: Selecciona entre modo comandos y modo datos, R/W: Selecciona entre modo lectura y modo escritura y EN: Habilita el Display.
- Bit 3: Este terminal se configura como entrada, y sirve para leer el valor del campo que suministra el separador de sincronismos activo.
- Bit 4: Este terminal se configura como entrada de reloj para el contador TMR0. Su misión es la de detectar el comienzo de los campos , y su funcionamiento se comenta detalladamente en el apartado de contadores.
- Bit 5: Este terminal configurado como entrada se encuentra conectado a la salida de Burst del separador de sincronismos. En la practica esta señal no tiene función alguna, pero se conecta para futuras mejoras del sistema.

La Puerta B consta de 8 terminales, los cuales tienen una doble funcionalidad:

1) Bus de datos del Display: La primera función de la puerta B es la de servir como Bus de Datos para el Display. Para esta función los terminales 0 a 7 se configuran como salidas, correspondiéndose estos con los bits DB0 a DB7 del Bus de Datos del Display.
2) Control del Teclado Matricial: Mientras no se manda información al display la puerta B controla un teclado matricial de 5 teclas. Para esta función los terminales 0 a 3 se configuran como salidas y los terminales 4 a 7 se configuran como entradas. Para el control del teclado se conectan las resistencias de polarización internas, y se utiliza la capacidad de interrupción por cambio de estado en la puerta.

Por ultimo, la Puerta C consta de 8 terminales, y sus funciones son las siguientes:

- Bit 0: Esta patilla se configura como entrada de reloj para el contador TMR1. Su función es la de contar los impulsos de sincronismo de la señal de vídeo, y su modo de operación se vio en el apartado correspondiente al TMR1.
- Bit 1: El terminal se configura como salida, y se destina al control del conmutador analógico. En el software se le llama salida de disparo.
- Bit 2: Este terminal configurado como salida se utiliza para el control del Buzzer. En el software se le llama salida buzzer.
- Bits 3 a 7: Estos terminales no tienen asignada ninguna función, pudiéndose utilizar para futuras ampliaciones y mejoras del sistema.

A continuación, se describen las características y modo de operación de todos los dispositivos conectados a las puertas del microcontrolador.

4.2.1.4.1. Display LM041L.

Se trata de un módulo visualizador LCD estándar, que se compone de una pantalla de cristal líquido (LCD), y un microcontrolador que la gobierna.

La pantalla consta de una matriz de 80 caracteres de 5x7 píxel organizados en 4 filas de 16 caracteres cada una (los últimos caracteres no son visibles). La inteligencia la proporciona cómo no, un microcontrolador incrustado con la pantalla, siendo el modelo el Hitachi 44780.

Características generales.

Las características generales del módulo LCD son las siguientes:

- Muestra caracteres ASCII, Japoneses, Griegos y símbolos matemáticos.
- Desplazamiento de los caracteres hacia la izquierda o la derecha.
- Proporciona la dirección de la posición absoluta o relativa del carácter.

- Memoria de 40 caracteres por línea de la pantalla (el display realmente consta de 2 filas de 40 caracteres organizadas como 4 filas de 16 caracteres).
- Movimiento del cursor y cambio de su aspecto.
- Permite que el usuario pueda programar 8 caracteres.
- Conexión a un procesador usando un interfaz de 4 u 8 bits.

Adaptación de la pantalla LCD.

El módulo LCD tiene 14 patitas, cuya descripción se muestra en la tabla 4.5. La alimentación es de +5V, y la regulación del contraste se realiza mediante voltaje obtenido al dividir los +5V con un potenciómetro de 10K.

La pantalla puede conectarse directamente a un microprocesador o a un microcontrolador mediante un bus de 4 u 8 bits (en nuestro caso el bus es de 8 bits).

Se precisan de 11 líneas en el modo de 8 bits. Las tres líneas de control son EN (Habilitación), I/D (Instrucción/Dato) y R/W (Lectura/Escritura). Las líneas de datos son triestado y pasan a estado de alta impedancia cuando la LCD no está activada.

El consumo del módulo es muy reducido (7.5mW) y su fácil manejo lo hacen un producto ideal para dispositivos que necesitan una capacidad de visualización pequeña o media.

Los valores máximos absolutos recomendados por el fabricante son los siguientes:

DESCRIPCIÓN		SIMBOLO	VALOR			UNIDAD
			Mínimo	Típico	Máximo	
Tensión de Alimentación		V_{DD} – GND	- 0.3	-	13.0	V
Tensión de Entrada		V_I	- 0.3	-	V_{DD}+0.3	V
Temperatura Ambiente	Operación	T_{OP}	- 20	-	+ 75	ºC
	Almacenaje	T_{STG}	- 55	-	+ 125	ºC

Tabla 4.5 – Descripción de terminales del módulo LCD.

Patilla número	Símbolo	Función
01	Vss	Masa
02	Vdd	+5V
03	Vee	Ajuste contraste
04	RS	Selección modo
05	R/W	Lectura / Escritura
06	EN	Validación
07	DB0	1ª Línea de datos LSB
08	DB1	2ª Línea de datos
09	DB2	3ª Línea de datos
10	DB3	4ª Línea de datos
11	DB4	5ª Línea de datos
12	DB5	6ª Línea de datos
13	DB6	7ª Línea de datos
14	DB7	8ª Línea de datos MSB

Comandos de la pantalla LCD.

Los comandos normalizados del microcontrolador HD44780 se resuman en la tabla 4.6. Como aclaración preliminar es importante destacar los siguientes puntos:

1) Cada operación se realiza cuando es activada la señal de habilitación EN.
2) Los tiempos indicados son los tiempos mínimos necesarios para realizar la operación en cuestión. Si no se tienen en cuenta estos tiempos se corre el riesgo de cometer errores en la representación de los caracteres en pantalla.

Tabla 4.6 – Comandos normalizados del microcontrolador HD44780.

INSTRUCCIÓN	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	DESCRIPCIÓN
Borrado	0	0	0	0	0	0	0	0	0	1	Borra la pantalla. Vuelve el cursor a la posición inicial (1.64ms)
Posición inicial	0	0	0	0	0	0	0	0	1	X	Vuelve el cursor a la posición inicial (40µs).
Modo	0	0	0	0	0	0	0	1	I/D	X	Controla de forma automática las direcciones de la RAM INC/DEC y si el visualizador se desplaza durante la escritura (40 µs).
Visualizador encendido/ apagado	0	0	0	0	0	0	1	D	C	B	Controla el visualizador encendido/apagado, controla el cursor activado o desactivado, controla el parpadeo del cursor. (40 µs).
Desplazamiento del cursor o de la pantalla.	0	0	0	0	0	1	S/ C	R/ L	X	X	Desplaza el cursor y/o el visualizador sin cambiar la RAM de visualización.
Selección de función.	0	0	0	0	1	D/ L	N	F	X	X	Selecciona el interfaz a 4 u 8 bits, ajusta el número de líneas de visualización, y selecciona la fuente de caracteres.
Selección de la dirección de la RAM del GC.	0	0	0	1	Dirección (CG).						Selecciona la dirección para la posterior escritura en la RAM del generador de caracteres. (40 µs).
Selección de la dirección de la RAM del visualizador.	0	0	1	Dirección (DD).							Selecciona la dirección para la posterior escritura en la RAM de visualización DDRAM (40 µs).
Lectura de la bandera de ocupado y la dirección.	0	1	BF	Contador de direcciones							Lectura de la bandera de ocupado (BF) y el valor actual del contador de dirección. (0 µs).
Escritura de los datos en la RAM del GC o del visualizador.	1	0	Los datos son escritos en la memoria RAM de visualización o en la del GC, en función del último mandato de control “SET DD/CG RAM ADDRESS”.								
Lectura de datos desde la RAM del GC o del visualizador	1	1	Los datos leídos de la RAM de visualización o de la del GC en función del último mandato de control “SET DD/CG RAM ADDRESS”.								Lee los datos desde el HD44780 (40 µs).

Donde:

I/D	0: Decrementa el puntero después de cada operación de lectura/escritura. 1: Incrementa el puntero después de cada operación de lectura/escritura.
S	0: Deja el cursor fijo 1: Desplaza el cursor con la visualización.
D	0: Display apagado. 1: Display encendido.
C	0: Cursor desactivado. 1: Cursor activado.
B	0: Cursor sin parpadeo. 1: Cursor con parpadeo.
S/C	0: Desplaza la visualización. 1: Desplaza sólo el cursor.
R/L	0: Las operaciones de desplazamiento se realizan a la izquierda. 1: Las operaciones de desplazamiento se realizan a la derecha.
DL.	0: Interface de 4 bits. 1: Interface de 8 bits.
N	0: Visualización a 1 línea. 1: Visualización a 2 líneas.
F	0: Fuente de caracteres de 5x7 puntos. 1: Fuente de caracteres de 5x10 puntos.
BF	0: No ocupado. 1: Ocupado.

Direccionado.

La RAM de las pantallas LCD no tiene un direccionamiento continuo y lineal pues el mapa depende de los caracteres y líneas que tenga el módulo.

Cada línea de la pantalla LCD tiene 40 caracteres de RAM de pantalla de datos (DDRAM) asociados. Las 40 posiciones de RAM están organizadas con un buffer de forma circular de tal forma que la última posición enlaza con la primera.

En nuestro caso, las dos líneas de 40 caracteres se encuentran divididas en cuatro líneas de 16 caracteres (las ultimas posiciones no son visibles). La distribución de las posiciones de memoria en la pantalla se muestra el la tabla 4.7.

Tabla 4.7 – Distribución de la memoria en la pantalla.

Línea	Posición carácter	Dirección DDRAM
1	00 – 15	80 – 8F
2	00 – 15	C0 – CF
3	16 – 39	90 – A8
4	16 – 39	D0 – D8

Modo de operación.

Cuando se enciende la LCD se produce un reset y esta queda a la espera de instrucciones. Normalmente, estas instrucciones encienden la pantalla y el cursor, y configuran la LCD para escribir de izquierda a derecha.

Hay que tener en cuenta que el controlador de la LCD tarda de 40 a 120us en completar una lectura o una escritura. Otras operaciones son más lentas y alcanzan hasta 5 ms.

A continuación se presentan las rutinas encargadas del manejo de la pantalla LCD. Estas rutinas son las de inicialización, escritura de comandos y escritura de datos.

```
void lcd_setup(void)          //configura e inicializa lcd
{
 delay_ms(50);                //espera encendido de lcd
 clear_bit(porta,lcd_en);     //mantiene lcd desabilitada
 clear_bit(porta,lcd_rw);     //mantiene modo escritura
 lcd_coman(system_set);       //envia comando interfaz 8 bits, pantalla 4 lineas, fuente 5x7
 lcd_coman(display_on);       //envia comando pantalla on
}
```

Figura 4.38 – Rutina que inicializa y configura la LCD.

```
void lcd_coman(char com)          //envia comando a lcd
{
 clear_bit(porta,lcd_rs);         //selecciona modo comandos
 trisb = 0x00;                    //configura puerta b para bus lcd (bits 7-0 out)
 lcd_write(com);                  //escribe comando
 delay_ms(5);                     //espera por si lcd ocupada
 set_bit(porta,lcd_rs);           //mantiene modo datos
}
```

Figura 4.39 – Rutina que envía comandos a la LCD.

```
void lcd_write(char dat)          //escribe comando o dato en la puerta b
{
 portb = dat;                     //escribe en la puerta b
 delay_ms(1);                     //espera por si lcd ocupada
 set_bit(porta,lcd_en);           //habilita lcd
 clear_bit(porta,lcd_en);         //deshabilita lcd
}
```

Figura 4.40 – Rutina que escribe comandos o datos en la LCD.

En la figura 4.38 se muestra la rutina que inicializa la LCD. Las operaciones que se realizan son:

1) Se espera 50ms para garantizar que se ha producido el reset y la estabilización de la pantalla LCD.
2) Normalmente se utiliza el PORTB para el control del teclado matricial, así que mientras no se realicen cambios en la pantalla esta se mantiene deshabilitada.
3) Como no se realizan lecturas, siempre se mantiene el modo de escritura.
4) Se envia el comando que configura la pantalla. La pantalla se configura para interfaz de 8 bits, 4 líneas y caracteres de 5x7 puntos.
5) Por ultimo se enciende la pantalla.

En la figura 4.39 se muestra la rutina que envia los comandos a la pantalla.

La pantalla se configura por defecto para enviar datos (debido a que se envían más datos que comandos). La rutina que envia los comandos ha de cambiar al modo comandos y luego otra vez al modo datos. La secuencia de instrucciones es la siguiente:

1) Selecciona modo comandos.
2) Configura el PORTB como salidas para el bus de datos.
3) Escribe el comando.
4) Espera ha que la LCD acepte el comando.
5) Por último vuelve al modo datos.

Por último, en la figura 4.40 se muestra la rutina que escribe los comandos o datos en la pantalla. Esta rutina es realmente la que envia la información a la pantalla LCD. La rutina consta de las siquientes instrucciones:

1) Envia al PORTB la información correspondiente al dato o comando.
2) Espera ha que la pantalla este disponible.
3) Por ultimo, se genera un pulso de habilitación para que acepte la información.

Para realizar una modificación en la pantalla se ha de mandar un comando obligatoriamente. Por ejemplo, antes de escribir un nuevo carácter en una determinada posición de la pantalla hay que situar el cursor en esa posición. Por este motivo el PORTB siempre se configura antes de mandar un comando.

Como se puede observar, para mandar datos a la pantalla (por ejemplo el siguiente carácter de una línea) se utiliza directamente la función lcd_write, ya que antes se ha tenido que mandar un comando y por tanto se habrá configurado el PORTB para el bus de datos.

4.2.1.4.2. Teclado Matricial.

Este tipo de teclados están configurados como una matriz filas-columnas con la intención de reducir el número de líneas de entrada y salida necesarias para conectarlo con el microcontrolador. En un teclado no matricial cada tecla necesita una línea de entrada.

Un teclado matricial está organizado de tal forma que cada tecla se conecta a una fila y una columna. El número de líneas de entrada necesarias para la matriz del teclado es igual a la suma de columnas y filas. El número de teclas que pueden conectarse a la matriz es el producto de las filas por las columnas (Figura 4.41).

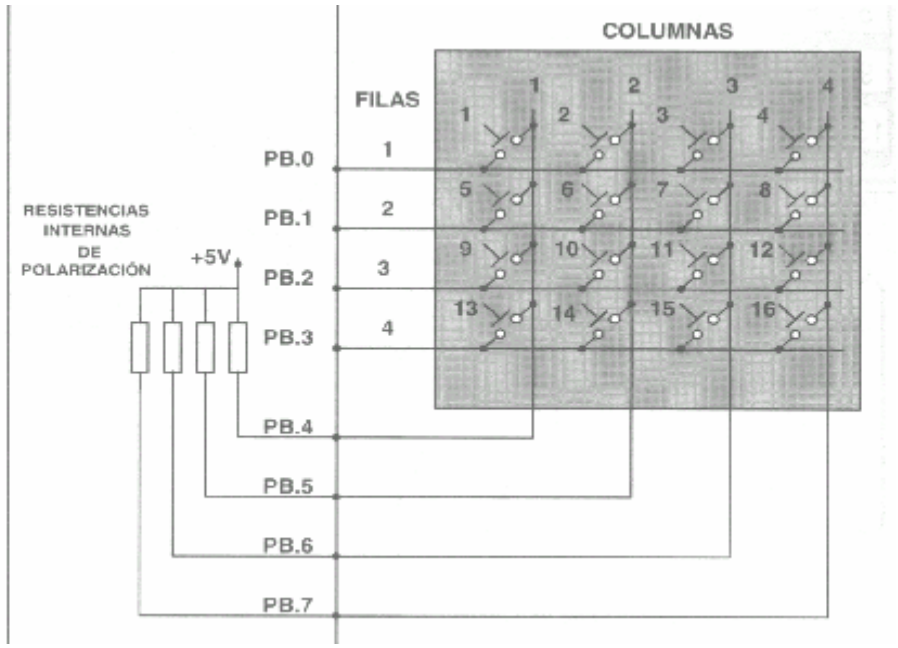


Figura 4.41 – Teclado matricial, mostrando la disposición interna en filas y columnas y su conexionado a las líneas de la Puerta B del microcontrolador.

Adaptación del teclado matricial.

En nuestro caso el teclado solo consta de cinco teclas, estando estas distribuidas de la siguiente forma:

1) Los terminales RB0 a RB3 se configuran como salidas para alimentar a las filas primera a cuarta respectivamente. En nuestro caso solo se utilizan las tres primeras salidas para alimentar a las tres primeras filas de la matriz. Las teclas 1ª y 2ª se conectan a la primera fila, las teclas 3ª y 4ª se conectan a la segunda fila, y la 5ª tecla se conecta a la tercera fila.
2) Los terminales RB4 a RB7 se configuran como entradas para leer las columnas primera a cuarta respectivamente. En este caso se utilizan todas las entradas, ya que son necesarias todas las columnas. La 1ª tecla se conecta a la primera columna, la 2ª tecla se conecta a la segunda columna, la 3ª tecla se conecta a la tercera columna, y por último las teclas 4ª y 5ª se conectan a la cuarta columna.

Tal como puede apreciarse en la figura 4.41, para conectar cinco teclas hubiera bastado una matriz de dos filas. Pero pensando en futuras ampliaciones, se decidió incorporar una tercera fila, posibilitando de esta forma la conexión de hasta 12 teclas. Ya que se disponen de tres filas, se hizo un reparto de las teclas entre las mismas, consistente en conectar las teclas 2ª y 3ª en la segunda fila.

Con la intención de proteger que dos salidas de la Puerta B sean cortocircuitadas por la pulsación de una tecla, cada fila incluye una resistencia en serie de 2.2KΩ.

Modo de operación.

La técnica de programación usada para gobernar una matriz de teclas necesita tanto entradas como salidas. Las filas están conectadas a las patitas de salida y las columnas a las de entrada.

Se comienza colocando a `0' la primera fila y las restantes a `1' . Si una tecla es pulsada en la columna `0', el `0' lógico aparece en la intersección fila-columna. Las columnas son exploradas de forma secuencial comprobando si hay un `0'. Si no se encuentra un `0', se pone a `0' la fila siguiente y la anterior a `1' pasando a comprobar nuevamente las columnas.

Si no es pulsada ninguna tecla en una fila las entradas se encuentran en estado flotante, razón por la que son necesarias las resistencias de polarización internas, que mantienen las entradas a nivel alto. Las resistencias programables internas permiten conectar el teclado matricial sin añadir componentes externos.

El programa también podría sacar cero en las columnas y comprobar las filas para detectar si se ha pulsado alguna tecla. No se hace así porque las columnas del teclado están conectadas a la Puerta B, que posee capacidad de interrupción.

Cuando se configuran como entradas los bits 4-7 de la Puerta B, pueden programarse para generar una interrupción en respuesta a un cambio de nivel de cualquiera de ellas respecto al último nivel leído. Las banderas que los señalizan y habilitan son RBIF y RBIE respectivamente.

La interrupción por cambio de estado en la Puerta B está idealmente diseñada para verificar la pulsación de una tecla mientras el PIC puede permanecer ejecutando cualquier otra tarea. Usando los bits 4-7 de la Puerta B como entradas de la matriz del teclado, si se pulsa una tecla, se genera una interrupción.

A continuación, en la figura 4.42 se muestra la rutina que da servicio al teclado matricial, en la cual se realizan las siguientes tareas:

1) Después de la pulsación, se espera 15ms para eliminar los rebotes de contacto.
2) Se lee el estado de los Bits 4 a 7 de la Puerta B para determinar la tecla pulsada.
3) Si la tecla pulsada está entre la primera y la tercera ambas inclusive, se comprueba el modo de funcionamiento, y dependiendo de la tecla y el modo se realizará una determinada función. Si por el contrario la tecla pulsada es la cuarta o la quinta, se procederá con el método de exploración del teclado comentado anteriormente. Una vez se determina si ha sido la 4ª o 5ª tecla, se procede de la misma forma que en el caso anterior.
4) Por último, después de soltar el pulsador, se espera 20ms para eliminar los rebotes de contacto y se prepara la Puerta B para una nueva interrupción.

```
//RSI Cambio de estado en puerta b
void portb_int(void)
{
 delay_ms(15);              //retardo para eliminar rebotes de contacto
 key = portb & 0xF0;        //lee estado puerta b para determinar tecla pulsada
 switch(key)                //determina tecla pulsada
 {
 case 0xE0:                 //1ª tecla pulsada
        if(modo == 0)       //si está en modo 0
         menu_1();          //va a modo 1
        else                //si está en cualquier otro modo
         menu_p();          //vuelve a modo 0
        break;              //retorna
 case 0xD0:                 //2ª tecla pulsada
        if(modo == 0)       //si está en modo 0
         menu_2();          //va a modo 2
        else                //si está en cualquier otro modo
         c_campo();         //cambia de campo
        break;              //retorna
 case 0xB0:                 //3ª tecla pulsada
        if(modo == 0)       //si está en modo 0
         menu_3();          //va a modo 3
        else                //si está en cualquier otro modo
         s_linea();         //sube línea
        break;              //retorna
 case 0x70:                 //4ª ó 5ª tecla pulsada
        portb = 0x04;       //bit 3 a 1 para determinar pulsador
        key = portb & 0xF0; //lee estado puerta b para determinar tecla pulsada
        if(key == 0x70)     //si entrada sigue a 0 -> 4ª tecla pulsada
        {
         if(modo == 0)      //si está en modo 0
          menu_4();         //va a modo 4
         else               //si está en cualquier otro modo
          b_linea();        //baja línea
        }
        else                //si entrada cambia a 1 -> 5ª tecla pulsada
         act_suma();        //actualiza sumador
 }
 delay_ms(20);              //retardo para eliminar rebotes de contacto
 trisb = 0xF0;   //configura puerta b para teclado(bits 7-4 entradas, 3-0 salidas)
 portb = 0x00;              //prepara puerta b para interrupción(bits 3-0 a 0) y
 asm movf _portb,w          ;actualiza w para comparación
}
```

Figura 4.42 – RSI que gobierna el teclado matricial.

4.2.1.4.3. Buzzer.

El buzzer proporciona una alerta sonora (pitido) cuando se dan cambios importantes en el sistema. Estos cambios que motivan el funcionamiento del buzzer son:

- Con Doble Pitido: Cuando tras conectar la alimentación, el microcontrolador consigue arrancar y se empiezan a ejecutar las instrucciones.
- Con Pitido Simple:

1) Cuando tras mostrarse los mensajes de inicio, se presenta el menú principal y el equipo comienza a funcionar (se comienza a disparar) .
2) Al entrar en cualquier modo de funcionamiento y al salir del mismo.
3) Cuando se cambia de campo, bien sea de forma automática o manual.
4) En los modos Vídeo y Continuo, al cambiar el rango para el salto de línea.

Modo de operación.

En la figura 4.43 se muestra el esquema del circuito para el buzzer. Como puede comprobarse, se ha optado por la simplicidad del circuito a consta de una mayor complejidad del software.

El circuito consiste en un simple seguidor de tensión el cual, gracias a su baja impedancia de salida, puede manejar sin problemas a un altavoz miniatura de 8Ω.

La resistencia de base (Rb) limita la corriente de salida por la patilla RC2 de la Puerta B, y al mismo tiempo limita la máxima corriente a través del altavoz.

Las características del altavoz son las siguientes:

- Impedancia = 8Ω.
- Potencia máxima = 300mW.

A partir de las características del altavoz se obtiene que la máxima tensión que puede aplicarse a este es de:

$$V = \sqrt{P \bullet R} = \sqrt{0.3 \bullet 8} = 1.55V.$$

Entonces, para evitar la rotura del altavoz, se fija la tensión límite en 1V.

Para 1V de tensión en el altavoz, por este circulará una corriente de aproximadamente:

$$Ic = \frac{1V}{8\Omega} = 125mA.$$

El transistor elegido es el BC337, y sus características son las siguientes:

- Transistor de Silicio – NPN.
- Vce = 50V, Ic = 0.8A, P = 0.625W, 100MHz.
- hFE = 290.

Entonces, la corriente de base será aproximadamente:

$$Ib = \frac{Ic}{hFE} = \frac{125mA}{290} = 432uA.$$

Suponiendo una tensión de salida de 5V en la patilla RC2 destinada al buzzer, una caída de tensión de 0.7V en la base del transistor y una caída de tensión de 1V en el altavoz, la corriente de base vendrá expresada por la ecuación:

$$Ib = \frac{5V - 0.7V - 1V}{Rb} = 432uA.$$

De donde se obtiene una resistencia de base Rb = 7632Ω, la cual se aproxima escogiendo una resistencia de 10KΩ. Con esta resistencia y el transistor antes mencionado se garantiza la integridad del microcontrolador y el altavoz.

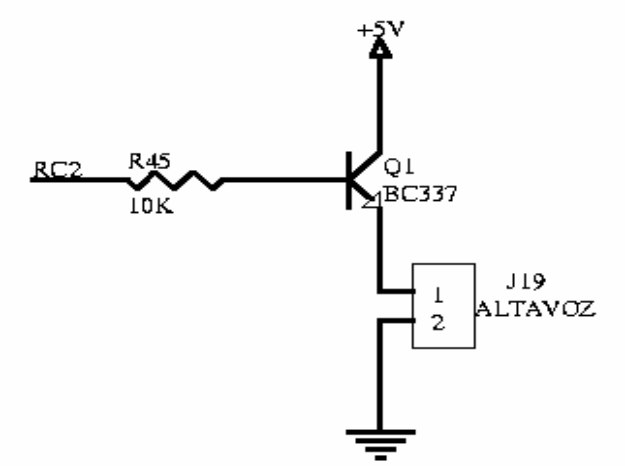


Figura 4.43 – Circuito destinado al control del buzzer.

Por último, se muestra el código del buzzer. Este es realmente el que genera el pitido. Para conseguirlo construye una señal rectangular de frecuencia apropiada, y lo hace cambiando 200 veces el nivel de salida en la salida RC2.

```
void buzzer(void)              //buzzer
{
 char i = 0;                   //define variable para el contar
 for(i=0; i<100; i++)          //señal rectangular para alerta sonora
 {
  set_bit(portc,2);
  delay_us(250);
  clear_bit(portc,2);
  delay_us(250);
 }
}
```

Figura 4.44 – Software destinado al buzzer.

4.2.2. SEPARADOR DE SINCRONISMOS ACTIVO LM1881N.

4.2.2.1. Descripción general.

El Separador de Sincronismos de Vídeo LM1881N extrae la información de tiempos, incluyendo Sincronismo Compuesto y Vertical, Burst/Portico Posterior e Información de Campo Par/Impar, de una señal de vídeo estándar con sincronismos NTSC, PAL o SECAM y amplitud de 0.5 a 2Vpp. El circuito integrado también es capaz de suministrar separación de sincronismos para señales de vídeo no estándar con barrido horizontal rápido.

Las salidas del LM1881N pueden usarse para capturar señales de vídeo cámara y reproductores de vídeo, suministrando información de los campos de vídeo para su almacenamiento en memoria, recuperar señales con el sincronismo suprimido o contaminado, y proveer de referencias de tiempo para la extracción de datos codificados o no codificados en líneas de vídeo específicas.

4.2.2.2. Características técnicas.

Las principales características suministradas por el fabricante son las siguientes:

Características principales.

- Señal de entrada compuesta acoplada en AC.
- Resistencia de entrada mayor de 10KΩ.
- Consumo de corriente menor de 10mA.
- Salidas de Sincronismo Compuesto y Vertical.
- Salida de Campo Par/Impar.
- Salida de Burst/Portico Posterior.
- Frecuencia de barrido horizontal de hasta 150KHz.
- Salida vertical con disparo rapido.
- Salida vertical con disparo por defecto para señales de vídeo no estándar (vídeo juegos y computadoras).

Características eléctricas.

- Suministro de corriente.....Vcc = 5V -> 5.2mA.
- Voltaje de entrada en DC.....1.3 a 1.8V.
- Voltaje umbral de entrada.....55 a 85mV.
- Corriente de descarga de entrada.....6 a 16uA.
- Voltaje de referencia en el terminal Rset.....1.1 a 1.35V.
- Salida de sincronismo compuesto.....0.2 - 4.5V.
- Salida vertical.....0.2 - 3.6V.
- Salidas burst y campo.....0.2 - 4.5V.
- Duración del sincronismo vertical.....230us.
- Duración del burst (2.7KΩ entre el terminal 5 y Vcc).....4us.
- Tiempo vertical por defecto.....65us.

Valores máximos absolutos.

- Suministro de voltaje.....13.2V.
- Voltaje de entrada.....Vcc = 5V -> 3Vpp.
  Vcc > 8V -> 6Vpp.
- Corriente de salida por terminales 1,3, y 5.....5mA.
- Corriente de salida por terminal 7.....2mA.
- Disipación de la cápsula.....1100mW.
- Rango de temperatura de operación.....0 a 70ºC.
- Rango de temperatura de almacenamiento.....-65 a 150ºC.
- Capacidad ESD.....2KV.
- Soldadura (DIL).....10s a 260ºC.

Tabla 4.8 - Descripción de terminales del separador de sincronismos de vídeo LM1881N.

Nombre	Número	Descripción
CSYNC	1	Salida de Sincronismo Compuesto.
VIN	2	Entrada de Vídeo Compuesto.
VSYNC	3	Salida de Sincronismo Vertical.
GND	4	Conexión a tierra.
BURST	5	Salida para Salva de Color.
Rset	6	Ajuste del Tiempo Vertical.
FIELD	7	Salida para Campo.
VCC	8	Entrada de Alimentación.

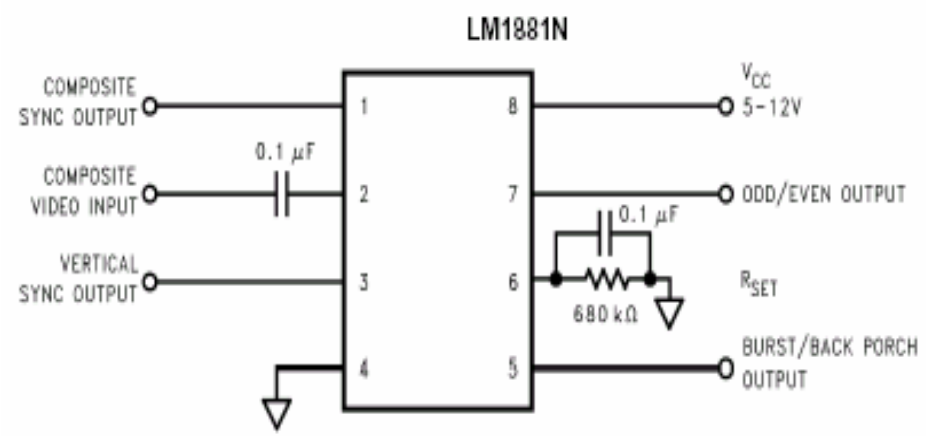


Figura 4.45 – Diagrama de terminales del circuito integrado LM1881N.

4.2.2.3. Conexión y modo de operación.

Conexión.

Se pueden adaptar señales de entrada con polaridad de vídeo positiva (incrementar el voltaje de la señal significa incrementar el brillo de la escena) de 0.5Vpp a 2Vpp.

El LM1881 opera con un suministro de voltaje positivo de entre 5 y 12VDC.

Los únicos componentes externos que se requieren, exceptuando el condensador de desacoplo en el terminal de alimentación y el condensador de desacoplo para el ajuste de corriente en el terminal Rset, son un condensador de desacoplo en la entrada de vídeo compuesto y una resistencia en el terminal Rset para ajustar el nivel de corriente interna. La resistencia en el terminal Rset permite al LM1881 ajustarse para fuentes de señal con frecuencias de barrido diferentes de 15.734KHz.

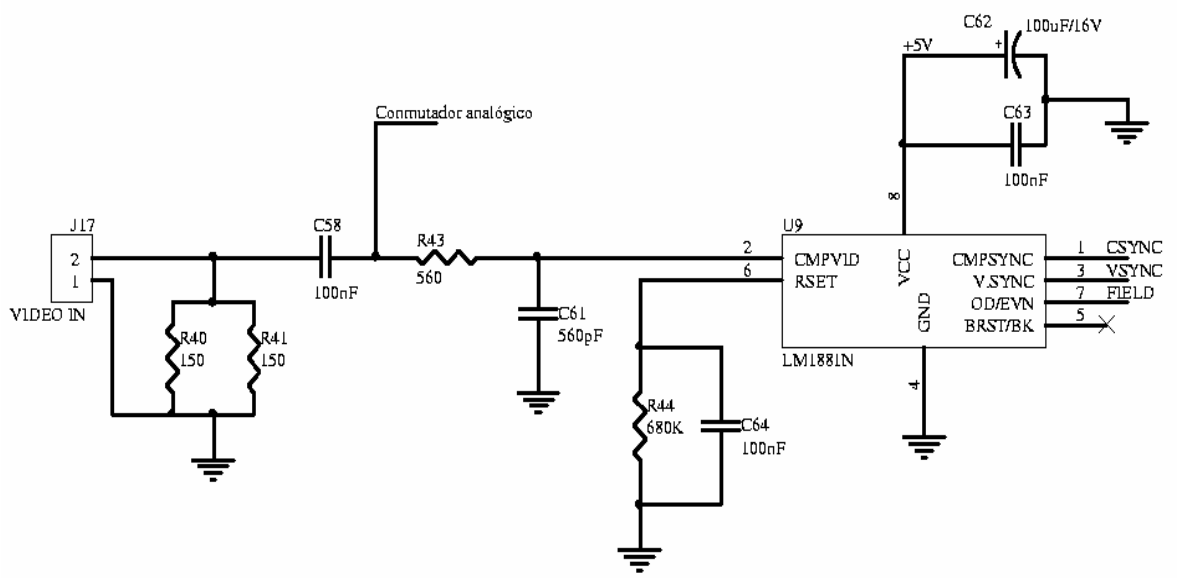


Figura 4.46 – Conexionado del separador de sincronismos en el monitor de vídeo.

De la figura 4.46 se extraen las siguientes observaciones.

La entrada de vídeo compuesto se conecta a una carga estándar de 75Ω, y luego pasa a través del condensador de desacoplo antes mencionado.

Después la señal sigue dos caminos. Por un lado se envía a la entrada del conmutador analógico, y por otro lado atraviesa un filtro paso bajo, también recomendado por el fabricante, antes de entrar en el separador. Con este filtro se elimina casi por completo la información de color, favoreciéndose así la operación del circuito integrado.

Por último, también pueden apreciarse la red de ajuste y desacoplo conectada al terminal Rset, así como la red de desacoplo conectada al terminal de alimentación.

Modo de operación.

La salida de sincronismo compuesto es, simplemente, una reproducción de la forma de onda de la señal de vídeo compuesto por debajo del nivel de negro. Esta señal se obtiene comparando la señal de entrada con un nivel de referencia apropiado.

La salida vertical se produce en el flanco de subida del primer serrado, en el periodo de sincronismo vertical. Después de un tiempo de retardo, se produce una salida vertical por defecto si el flanco ascendente mencionado no aparece dentro de un periodo de retardo que se establece externamente, como suele ocurrir en el caso de señales de vídeo no estándar.

La salida de campo identifica, que campo de una fuente de vídeo entrelazado está presente en la entrada. Para detectar los campos par/impar, el LM1881N integra la señal de sincronismo compuesto. Un condensador se carga durante el periodo entre pulsos de sincronismo, y se descarga cuando el pulso de sincronismo está presente. El periodo entre pulsos de sincronismo horizontal normales es suficiente para permitir al voltaje del condensador alargar el nivel umbral de un comparador que borra un flip-flop, el cual también esta sincronizado con la señal de sincronismo.

Por último, en la figura 4.47 se muestran todas las señales proporcionadas por el separador de sincronismos. Estas señales son:

a) Vídeo compuesto.
b) Sincronismo compuesto.
c) Pulso de salida vertical.

d) Indicador de campo par/impar.

e) Indicador de salva de color/portico posterior.

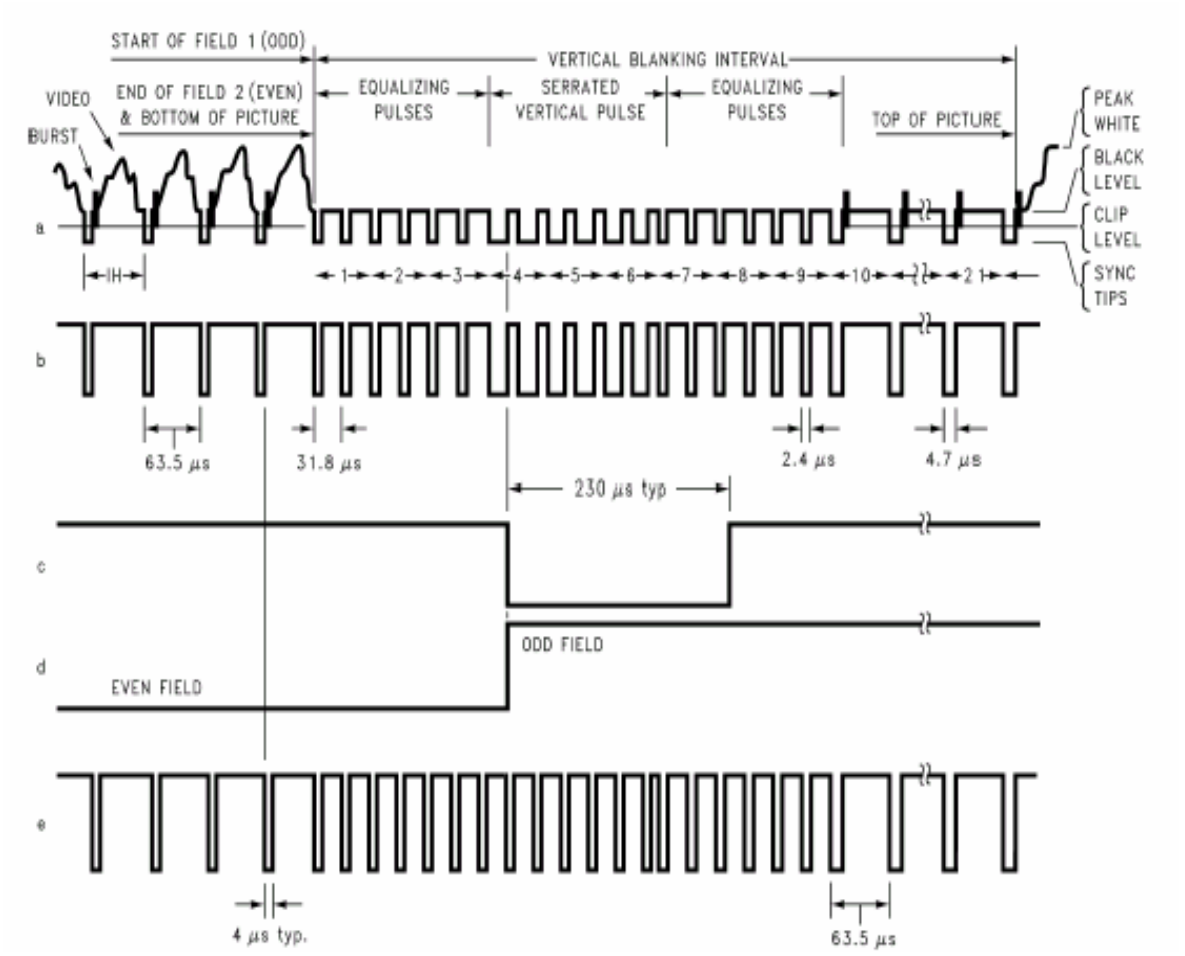


Figura 4.47 – Salidas suministradas por el circuito integrado LM1881N.

4.2.3. CONMUTADOR ANALÓGICO CD4066BC.

4.2.3.1. Descripción general.

El CD4066BC es un cuádruple conmutador analógico bilateral, diseñado para la transmisión o multiplexión de señales analógicas y digitales. Es compatible pin a pin con el CD4016BC, pero tiene una resistencia de encendido mucho menor, y la resistencia de encendido es una constante relativa sobre el rango de la señal de entrada.

Como aplicaciones directas se tienen:

- Conmutación / multiplexión de señales analógicas:
  - Modulación / Demodulación.
  - Troceador.
  - Silenciador.
  - Conmutador.
- Conmutación / multiplexión de señales digitales.
- Implementación lógica CMOS.
- Conversión analógica-digital y digital-analógica.
- Control digital de frecuencia, impedancia, fase y ganancia de señales analógicas.

4.2.3.2. Características técnicas.

Las principales características suministradas por el fabricante son las siguientes:

Características principales.

- Amplio rango del voltaje de operación -> 3 a 15V.
- Alta inmunidad al ruido -> 0.45Vdd.
- Amplio rango de conmutación analógica y digital -> +/- 7.5Vpp.
- Resistencia de encendido de 80Ω para operación con 15V.
- Variación de la resistencia de encendido en 5Ω como máximo.
- Resistencia de encendido constante para todo el rango de la señal.

- Alta relación de encendido/apagado del voltaje de salida -> 65dB.
- Alto grado de linealidad -> 0.1% de distorsión.
- Fuga del conmutador en apagado extremadamente baja -> 0.1nA.
- Impedancia de entrada de control extremadamente alta -> 100MΩ.
- Baja interferencia entre conmutadores -> -50dB.
- Respuesta en frecuencia en conducción de 40MHz.

Características eléctricas.

- Corriente de reposo .....1uA.
- Resistencia de encendido (para RL = 10kΩ y VDD = 5V).....270Ω.
- Variación de la resistencia de encendido.....10Ω.
- Fuga de entrada o salida sin conducción.....0.1nA.
- Nivel bajo del voltaje de entrada.....1.5V.
- Nivel alto del voltaje de entrada.....2.75V.
- Corriente de entrada.....+/- 0.3uA.
- Retardo de propagación de entrada a salida.....55ns.
- Capacidad en la entrada de señal.....8pF.
- Capacidad en la salida de señal.....8pF.
- Capacidad en la entrada de control.....5pF.

Valores máximos absolutos.

- Suministro de voltaje.....-0.5 a 18V.
- Voltaje de entrada.....-0.5 a VCC+0.5V.
- Temperatura de almacenamiento.....-65 a 150ºC.
- Disipación de potencia.....700mW.
- Soldadura.....10s a 300ºC.

Condiciones de operación recomendadas.

- Suministro de voltaje.....3 a 15V.
- Voltaje de entrada.....0 a VDD.
- Temperatura de operación.....-40 a 85ºC.

4.2.3.3. Conexión del circuito.

Por último, se muestran el esquema del circuito, el diagrama de conexión y el conexionado del circuito en nuestro sistema.

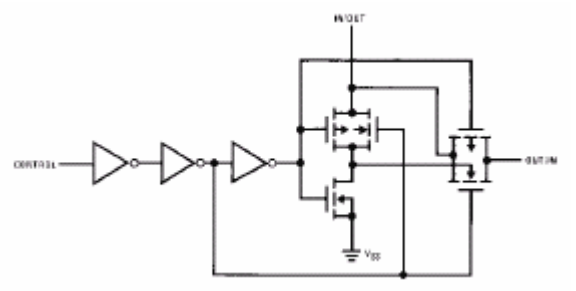


Figura 4.48 - Diagrama esquemático del conmutador analógico CD4066BC.

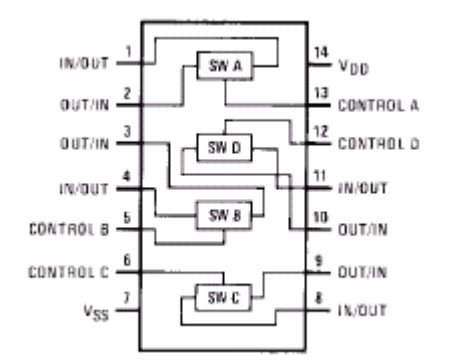


Figura 4.49 – Diagrama de terminales del conmutador analógico CD4066BC.

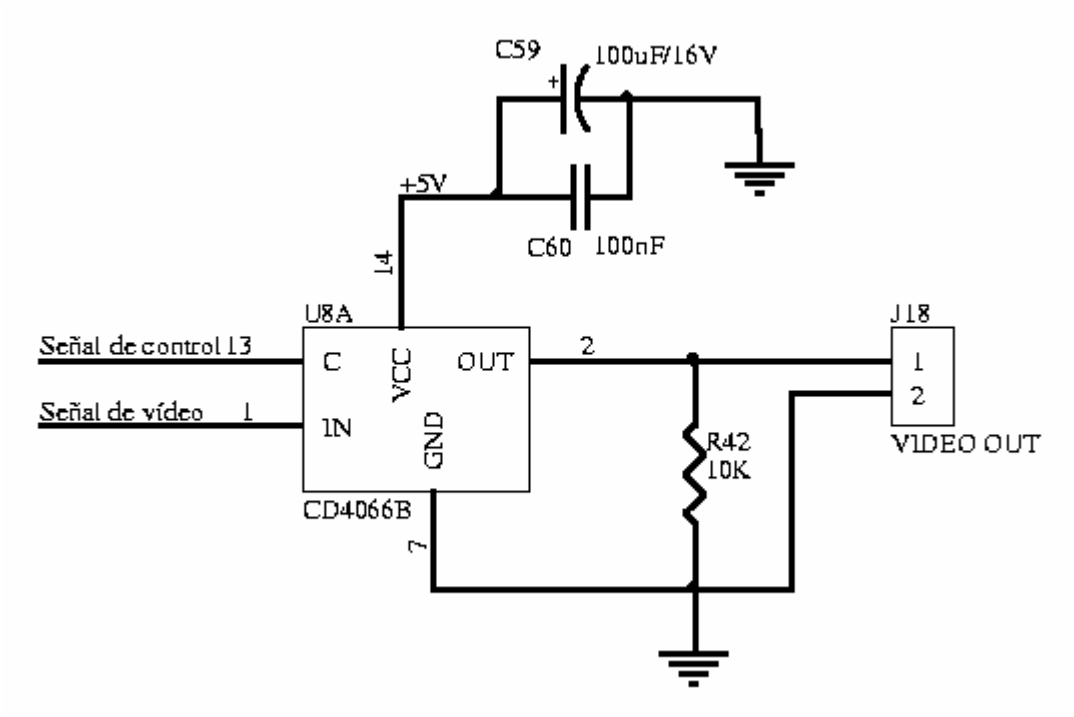


Figura 4.50 - Conexiones del conmutador en el monitor de vídeo.

En la figura anterior puede observarse que la circuitería necesaria para la conexión del circuito es mínima. Básicamente consiste en una carga de salida (el valor es el recomendado por el fabricante) y unos condensadores de desacoplo en la línea de alimentación (estos filtros se verán en el apartado de la fuente de alimentación).

4.3. FUENTE DE ALIMENTACIÓN Y FILTRADO.

FUENTE DE ALIMENTACIÓN Y FILTRADO

En la mayoría de los circuitos electrónicos es necesario la utilización de una o varias fuentes de energía eléctrica en forma de tensión continua estabilizada.

En lo referido al rendimiento de la fuente y calidad de la onda de salida, las fuentes de alimentación conmutadas son superiores a las lineales, pero también son más costosas y difíciles de diseñar. Por esto, debido al bajo consumo del sistema, junto con la necesidad de no complicar ni encarecer mucho el diseño, se opto por la utilización de una fuente de alimentación lineal tradicional.

Las fuentes de alimentación lineales pueden considerarse como convertidores estáticos que, a partir de una fuente de energía eléctrica primaria, proporcionan energía eléctrica continua con unos determinados requisitos de calidad.

La fuente de energía primaria suele ser la red eléctrica. En este caso se necesitarán convertidores estáticos que transformen la tensión alterna en continua, lo cual se consigue mediante rectificadores. La señal de salida del rectificador se mejorará con la utilización de un filtro adecuado, obteniéndose en su salida una componente continua y unos armónicos o rizado. Finalmente, para obtener diferentes niveles de tensión y asegurar que estros niveles se mantiene constantes ante variaciones de la tensión de entrada y de la corriente de salida, se utilizan unos circuitos denominados reguladores. El bloque formado por el transformador, el rectificador, el filtro y el regulador se suele denominar fuente de alimentación.

El esquema de bloques básico de una fuente de alimentación lineal alimentada a partir de la red eléctrica se muestra en la siguiente figura.

TRANSFORMADOR. RECTIFICADOR. FILTRO. REGULADOR.

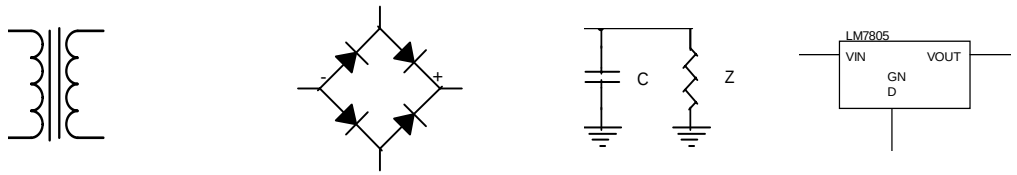


Fig. 4.51. - Diagrama de bloques básico de una fuente de alimentación lineal.

4.3.1. Requisitos.

El sistema debe ser autónomo, por lo tanto el circuito de alimentación debe estar incluido en la misma caja. También se exige que el sistema se monte por completo en una sola placa de circuito impreso, lo cual simplificará el procedimiento de fabricación y montaje.

La fuente de alimentación debe proporcionar energía a todos los componentes del sistema. Por lo tanto es esencial que ésta sea capaz de suministrar más energía de la que estos vayan a consumir. Las exigencias requeridas por el sistema en cuanto a la alimentación son las siguientes:

1) En el generador de vídeo ha de alimentar a los siguientes dispositivos:

   - Microcontrolador PIC16F84-10.
   - Oscilador con circuito integrado 74HC04N.
   - Conversor de vídeo CXA1645P.
   - Conversor de vídeo MC1377P.

Todos los dispositivos se alimentan con una tensión de +5V, excepto el circuito integrado MC1377 que requiere una alimentación de +12V.

2) En el monitor de vídeo se ha de alimentar a los siguientes dispositivos.

- Microcontrolador PIC16F876-04.
- Display LM041L.
- Buzzer.
- Separador de sincronismos de vídeo LM1881N.
- Conmutador analógico con circuito integrado CD4066BC.

En este caso todos los dispositivos requieren una tensión de alimentación de +5V.

Suponiendo despreciables las intensidades que derivan los reguladores de tensión hacia tierra, la intensidad que consume el circuito será, la suma de la intensidad que circula por la línea de alimentación de +12V, más la intensidad que circula por la línea de alimentación de +5V.

Según mediciones realizadas en los prototipos se tiene que:

1) Por la línea de alimentación de +12V (esta solo alimenta al CI MC1377) circula una intensidad de 50mA aproximadamente en el caso más desfavorable.
2) Por la línea de alimentación de +5V (esta alimenta al resto del circuito) circula una intensidad de 75mA aproximadamente cuando no funciona el buzzer, mientras que cuando funciona el buzzer la intensidad alcanza un valor máximo de 225mA aproximadamente.

Por tanto, los requisitos para la fuente de alimentación son:

1) Tensiones de salida de +5VDC y +12VDC, con una buena regulación y el menor rizado posible.
2) Intensidad de salida superior a 275mA eficaces.

4.3.2. Descripción del circuito.

El diseño de la fuente de alimentación no introduce ninguna solución novedosa, siendo este bastante tradicional y sencillo, pero no por ello deja de ser eficaz.

La simplicidad del circuito se debe en gran medida a la utilización de reguladores de tensión integrados.

Para comentar el circuito dividimos este en dos partes, una encargada de obtener tensión continua filtrada a partir de la tensión de red, y otra encargada de suministrar los diferentes niveles de tensión a partir de la tensión continua antes mencionada.

La primera parte del circuito se muestra a continuación en la figura 4.52.

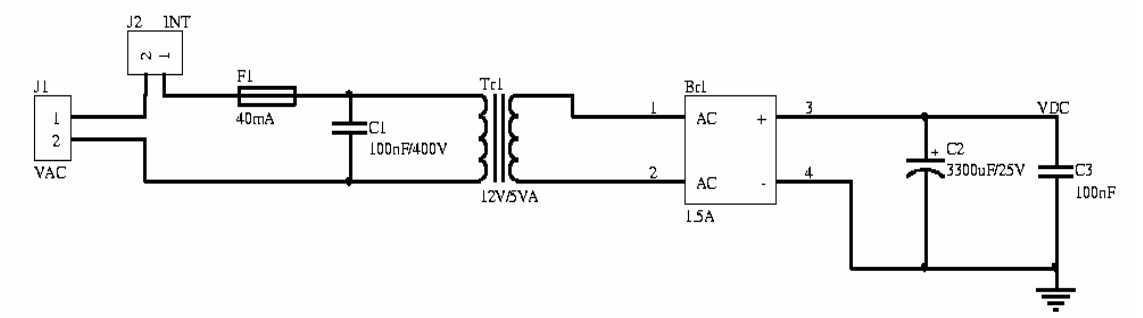


Figura 4.52 – Circuito que obtiene tensión continua filtrada a partir de la tensión de red.

Los bloques que constituyen esta primera parte son:

1) Conector de entrada: Donde se introduce la tensión de red.
2) Interruptor: Se conecta en serie un conector, que permite exteriorizar un interruptor de red para la conexión/desconexión del sistema.
3) Fusible: El transformador incluye un fusible de protección integrado en la cápsula, y en caso de que este saltara habría que sustituir el transformador completo. Por este motivo se incluye un fusible externo, cuya corriente máxima es menor que la del fusible integrado.

4) Filtro de red: Desacopla la tensión de red evitando pequeñas perturbaciones producidas por otros equipos también conectados a la red.
5) Transformador de red: Reduce el voltaje desde 220V eficaces hasta 12V eficaces. También proporciona aislamiento de red.
6) Puente rectificador B380C1500: Es un puente rectificador integrado, que soporta una tensión de 380V y una corriente de 1.5A.
7) Filtro: Compuesto por dos condensadores, uno de 2200uF para el desacoplo de la componente de baja frecuencia, y otro de 100nF para el desacoplo de la componente de alta frecuencia.

La segunda parte del circuito se muestra a continuación en la figura 4.53:

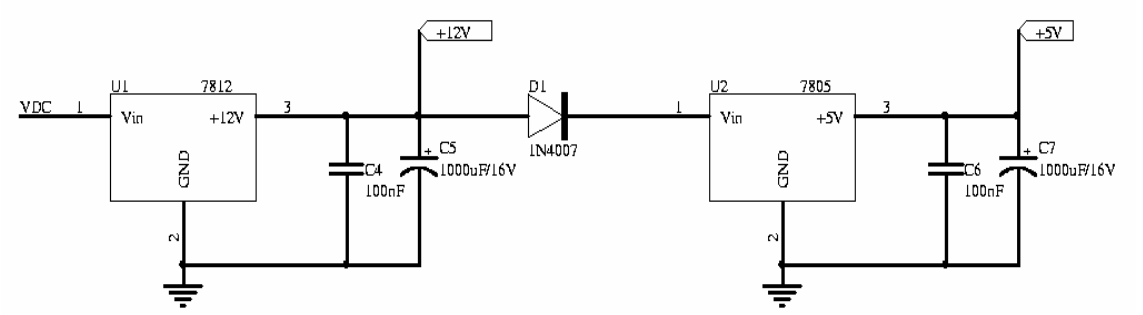


Figura 4.53 – Circuito que suministra las diferentes tensiones de alimentación.

Los bloques que constituyen la segunda parte son:

1) Regulador de tensión 7812: Este circuito integrado con cápsula TO220, es capaz de entregar una tensión fija regulada de 12V y una intensidad de hasta 1A. Dispone de protección interna contra sobrecargas, térmica y contra cortocircuitos. El rango de tensión de entrada va desde 14.5 a 35 voltios. Según la disipación de potencia será necesario montar un radiador adecuado en la cápsula.

2) Filtro para 12V: Compuesto por dos condensadores, uno de 1000uF para el desacoplo de la componente de baja frecuencia, y otro de 100nF para el desacoplo de la componente de alta frecuencia.
3) Diodo 1N4007: A través de el se alimenta al regulador que proporciona la tensión de 5V. Ayuda a reducir la tensión que se aplica al segundo regulador, reduciéndose de esta forma la disipación de potencia en la cápsula.
4) Regulador de tensión 7805: Tiene las mismas características que el 7812, solo que la tensión de salida es de 5V y el rango de tensión va desde 7.5 a 30 voltios.
5) Filtro para 5V: Es idéntico al utilizado para la tensión de 12V.

4.3.3. Cálculos.

El transformador utilizado dispone de un primario de 230V eficaces, y un secundario de 12V eficaces y una potencia de 5VA. Por tanto la intensidad eficaz que puede suministrar el transformador es aproximadamente:

$$Io = \frac{5VA}{12V} = 417mA.$$

Por tanto, la intensidad que puede suministrar el transformador es adecuada para cumplir con la especificación.

El puente rectificador soporta hasta 1.5A, por lo que está sobredimensionado.

Si el transformador proporciona 12V eficaces, a la salida del puente rectificador tendremos una tensión:

$$V_{dc}\ \text{máx.} = (12 \times \sqrt{2}) - 1.4 = 15.5\ \text{V}.$$

La tensión mínima en la entrada del regulador de 12V para un correcto funcionamiento del circuito (para que se produzca regulación) es de 14.5V, entonces la capacidad necesaria para el filtro será de:

$$C = \frac{Ic}{2 \times f \times Vr}$$

Ic : Intensidad.

F: Frecuencia.

Vr: Tensión de rizado.

Vdc máx. = 15.5V.

Vo = 14.5V (tensión de entrada del regulador).

V rizado = 15.5 – 14.5 = 1V.

Sustituyendo en la formula anterior se obtiene una capacidad de: C = 2750μF, y debido a que este valor no está normalizado, se opta por el valor normalizado de 3300uF.

Finalmente, también son necesarios unos filtros a las salidas de los reguladores de tensión.

Los valores recomendados por el fabricante (de 10uF a 100uF) no son tan elevados como los utilizados en el diseño, siendo esto debido a lo siguiente:

1) Para disminuir la disipación de potencia en el regulador de 5V, este se conecta en cascada con el regulador de 12V (se disminuye notablemente la caida de tensión).
2) Existen muchos circuitos integrados conectados a la salida del regulador de 5V.

4.3.4. Filtrado.

Este apartado se refiere a los filtros que incorpora cada circuito integrado en sus líneas de alimentación.

En el prototipo del generador se apreciaban deficiencias en la señal de salida, lo cual estaba causado por ruidos existentes en determinadas líneas de señal, y principalmente por el gran ruido existente en las líneas de alimentación.

Se realizaron mediciones del ruido en los terminales de alimentación de los circuitos integrados y en las isletas de cobre muerto (sin conexión). De los resultados obtenidos, los más significativos se muestran a continuación:

1) Señal de 4.43MHz y amplitud de hasta 200mV, en las líneas de alimentación y tierra del oscilador que alimenta al conversor CXA1645.
2) Señal de 4.43MHz y amplitud de hasta 160mV, en las isletas de cobre muerto alrededor del oscilador que alimenta al conversor CXA1645.
3) Señal de 10MHz y amplitud de hasta 6mV, en las isletas de cobre muerto alrededor del microcontrolador PIC16F84.
4) Señal de aproximadamente 16KHz y hasta 32mV, en las isletas de cobre muerto alrededor de los dos conversores de vídeo.

Teniendo en cuenta los resultados obtenidos en las mediciones se puede decir lo siguiente:

1) Los osciladores, circuitos con oscilador y circuitos que consumen mucha corriente, producen grandes oscilaciones en sus líneas de alimentación debido a saltos bruscos en la corriente que demandan. Las pistas demasiado largas y demasiado delgadas provocan importantes caídas de tensión.
2) Las pistas con señales de alta frecuencia o niveles de tensión elevados, inducen ruido en las pistas cercanas o separadas por isletas de cobre muerto. La separación de las pistas, su tamaño y su ubicación, pueden provocar acoplamientos inductivos y capacitivos para señales de determinada frecuencia y amplitud.

A continuación, para mejorar las deficiencias del generador se realizaron las siguientes modificaciones:

1) Se conectaron condensadores de 100uF y de 100nF, lo más cerca posible de los terminales de alimentación, en los circuitos más problemáticos.
2) Se conectaron todas las isletas posibles a la conexión de tierra más cercana.
3) Se conectó un conductor entre el punto de masa escogido como referencia y el punto de masa más alejado de este.

Tras las modificaciones, los mismos niveles mostrados en el apartado anterior se habían reducido hasta en un 75% en la mayoría de los casos.

Teniendo en cuenta todo lo expuesto, en el diseño de la placa de circuito impreso definitiva se cuidaron los siguientes aspectos:

1) Se colocaron filtros adecuados en cada uno de los circuitos, lo más cerca posible de sus terminales de alimentación.
2) Las pistas de alimentación se agrandaron en la medida de lo posible.
3) Se colocó un plano de tierra en la cara superior de la placa, a través del cual se conectan todos los puntos de masa.
4) Todas las isletas de cobre muerto también se conectaron al plano de tierra.
5) Los osciladores y las pistas de salida de los mismos se rodearon con grandes pistas de masa, y se alejaron en la medida de lo posible de pistas con señales críticas.

Por último, se muestran los circuitos de filtrado utilizados en cada uno de los casos.

Para los conversores de vídeo se recomienda la red de filtrado que se muestra a continuación, donde R1 representa la resistencia de la pista de alimentación.

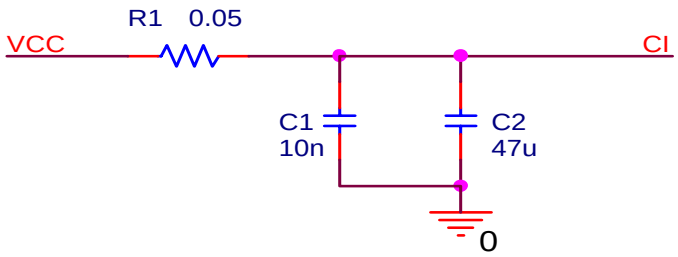


Figura 4.54 – Filtro recomendado para conversores de vídeo.

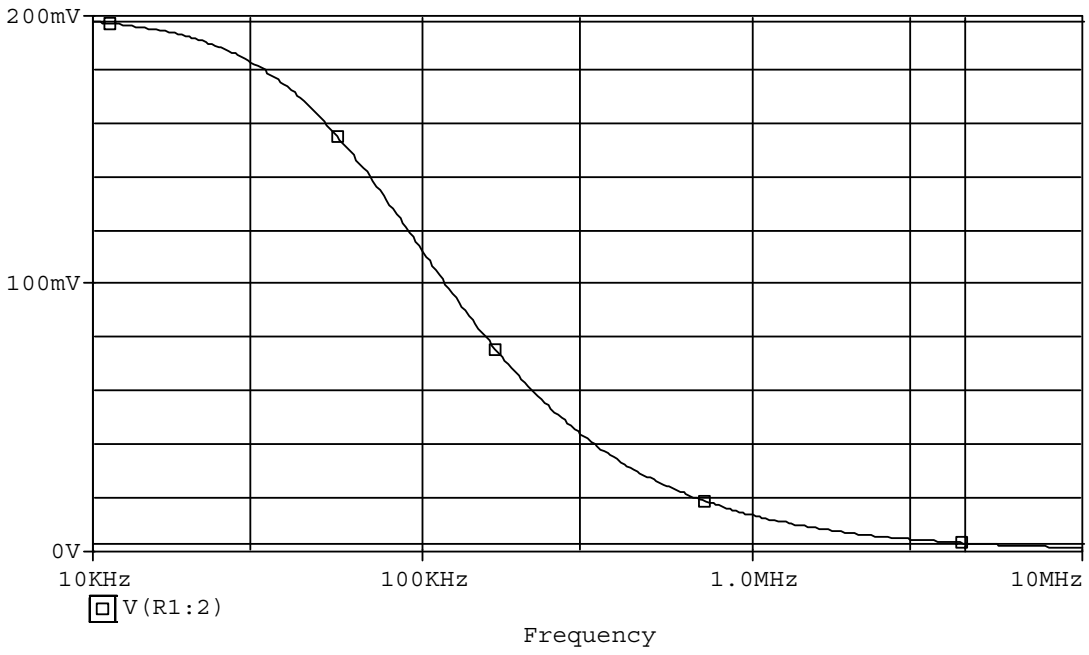


Figura 4.55 – Repuesta para el circuito de filtrado de la figura 4.54.

Para la simulación se han considerado los siguientes aspectos:

1) Un nivel de entrada de 200mV, ya que este era el máximo nivel de ruido encontrado en las mediciones anteriores.
2) Un barrido desde 10KHz hasta 10MHz, ya que este rango incluye todas las frecuencias de posibles ruidos.

En la figura anterior, el cursor de medida indica un nivel de 3mV para la frecuencia de 4.43MHz, lo cual significa que se produce una buena atenuación de la señal de ruido más problemática. No conforme, se intentó mejorar la atenuación, para lo cual se aumentó el orden del filtro tal como se indica en la figura siguiente.

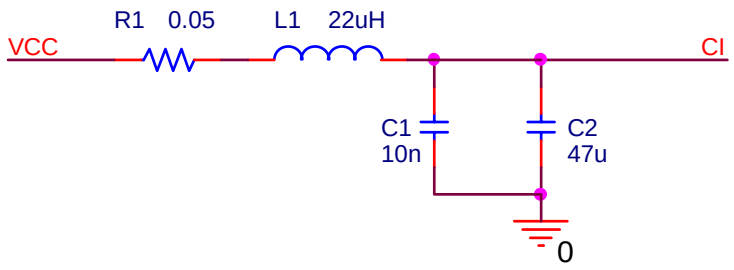


Figura 4.56 – Circuito modificado para aumentar la atenuación.

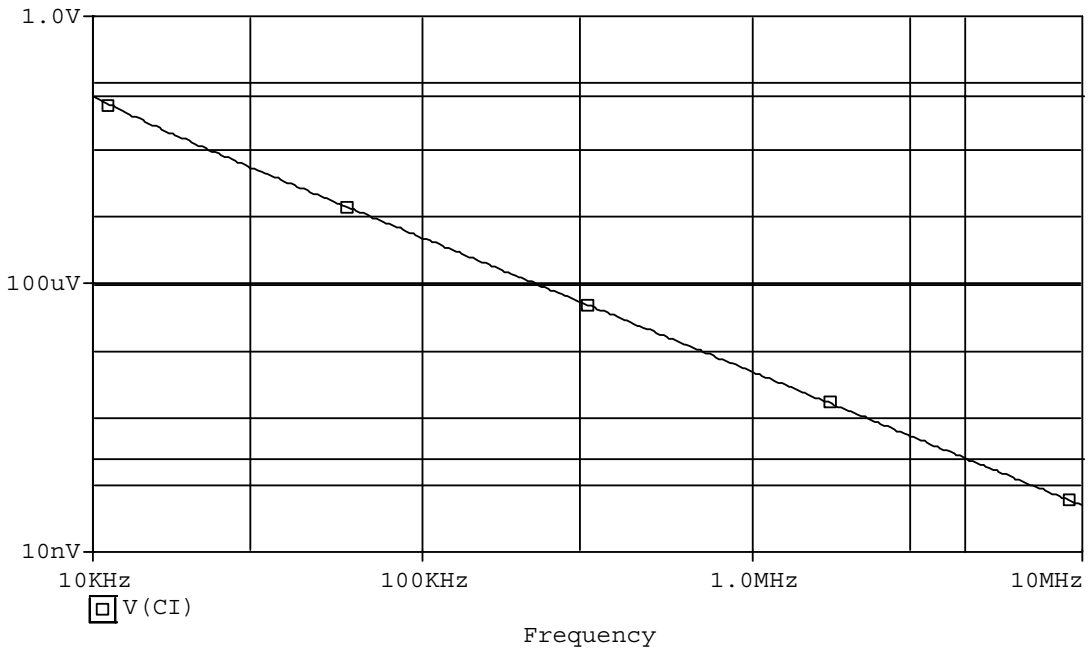


Figura 4.57 – Respuesta del circuito de filtrado modificado.

En la figura anterior, el cursor de medida indica un nivel de 0.25uV para la frecuencia de 4.43MHz. Por tanto se mejora la atenuación del filtro inicial recomendado, y este será el filtro adoptado para las líneas de alimentación de los conversores de vídeo.

Debido a que el oscilador de 4.43MHz produce grandes perturbaciones en la línea de alimentación, se decidió incorporar un buen filtro en dicha línea. Para ello se modificó el filtro utilizado para los conversores de vídeo, cambiando los condensadores por 100nF y 100uF respectivamente.

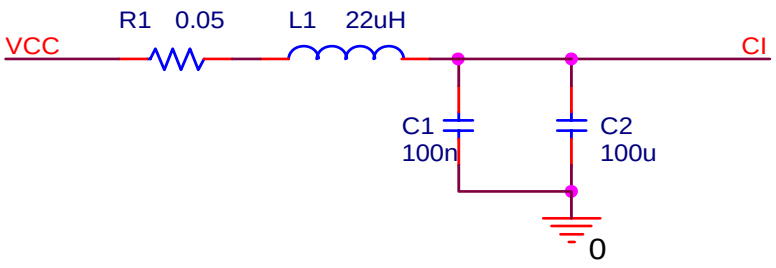


Figura 4.58 – Circuito de filtrado para el oscilador de 4.43MHz.

Para el resto de dispositivos el circuito de filtrado se corresponde con el mostrado en la figura 4.59, consistente en un desacoplo común.

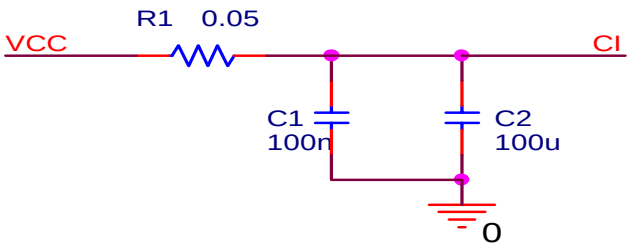


Figura 4.59 – Filtro de alimentación común.

CAPITULO 5:
SOFTWARE.

5.1. SOFTWARE DEL GENERADOR.

SOFTWARE DEL GENERADOR

Desde el comienzo del proyecto se enfocó la gestión del generador de vídeo con un microcontrolador. El microcontrolador PIC16F84-10 es el encargado de la gestión del sistema, y el control de todos los módulos conectados al mismo se realiza a través de su Programa de Aplicación.

El programa está realizado en Lenguaje Ensamblador. La programación en este lenguaje es muy lenta y laboriosa, pero nos vemos obligados a utilizarla por motivos de temporización. La temporización que se requiere para la base de tiempos de la señal de vídeo compuesto es muy exigente, y la única forma de conseguirla es con un control estricto del número de instrucciones que se ejecutan.

Para la creación del software se utilizó la herramienta de desarrollo proporcionada por el fabricante del microcontrolador. El fabricante es Microchip, y la herramienta de desarrollo se denomina MPLAB. Esta herramienta incluye:

- Un editor de texto, para la confección del código fuente.
- Un ensamblador, para la creación del código ejecutable.
- Un simulador, que permite comprobar el funcionamiento del software.

El formato empleado para almacenar el programa es de tipo hexadecimal ".HEX ".

El programa se almacena en la memoria EEPROM del mencionado microcontrolador utilizando para ello el programador de microcontroladores “TE-20”.

5.1.1. DIAGRAMA DE FUNCIONES.

En este apartado se verá, de forma esquemática, el proceso que sigue el programa de la aplicación.

El programa realiza las siguientes funciones:

1) Acepta por el terminal 2 del PORTA las órdenes provenientes de un pulsador, para de este modo poder seleccionar el patrón a generar.
2) Genera en el terminal 0 del PORTB todos los sincronismos requeridos por la norma de televisión adoptada.
3) Genera en los terminales 3, 4 y 2 las señales R, G y B respectivamente, que correspondan con el patrón que deba mostrarse a la salida.

Y es capaz de generar los siguientes patrones de vídeo:

1) Barras de color.
2) Barrido en blanco.
3) Barrido en Rojo.
4) Barrido en Verde.
5) Barrido en Azul.
6) Damero.
7) Rejilla.

El modo de operación es el siguiente:

1) Tras conectarse la alimentación, el sistema se resetea automáticamente.
2) Después del reset se inicia el sistema. En el inicio se configuran las puertas (entradas y salidas), y se establece el nivel de reposo en la línea de sincronismo y las líneas de información de color.
3) Seguidamente, se barre el primer patrón.
4) Al final del cuadro se comprueba si se ha pulsado el interruptor. Si se ha pulsado, se salta al siguiente modo de barrido, y en caso contrario, se continua barriendo el mismo patrón.

El proceso de exploración del teclado se repite al final de cada cuadro. De esta forma se va accediendo secuencialmente a cada uno de los modos de funcionamiento.

El diagrama de funciones principal es el siguiente:

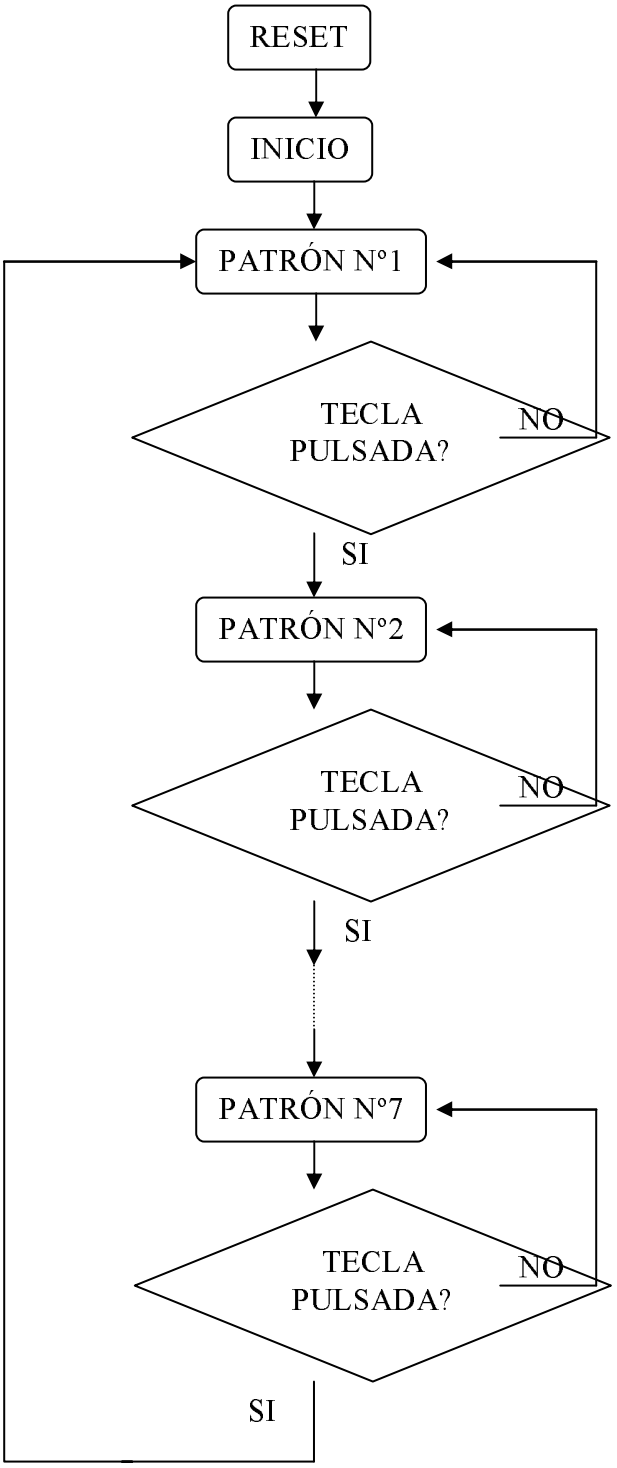


Figura 5.1 – Secuencia principal del software del generador de vídeo.

Dentro de cada modo de barrido se siguen unos pasos comunes. Estos pasos son los siguientes:

1) Inicio:
    a) Se configuran los bits que señalizan el modo de funcionamiento.
    b) Se configura el bit señalizador de campo, para indicar que se comienza con el barrido del campo impar.
    c) Se actualiza el contador de impulsos de igualación anterior.

2) Actualiza contadores: Se actualizan los contadores que controlan el número de impulsos de sincronismo vertical y de igualación posterior, el numero de líneas de negro y el número de líneas de vídeo.

3) Sincronismo vertical: Esta función genera los 5 impulsos de igualación anterior, los 5 impulsos de sincronismo vertical y los 5 impulsos de igualación posterior. El modo de operación de la función consiste en ir cambiando el nivel en la salida de sincronismo. Entre los cambios de nivel se introducen instrucciones de no operar y bucles de retardo, hasta que se consigue la temporización adecuada.

4) Líneas de negro: Esta función genera las líneas de negro que existen antes de las líneas de vídeo. En el inicio se configuraron las salidas de componentes de color para el nivel de negro, y durante el sincronismo vertical no se modifica este nivel. Por lo tanto, está función lo único que hace es generar el impulso de sincronismo horizontal, manteniendo el nivel de negro. El modo de operación es igual al de la anterior función.

5) Líneas de vídeo: A partir de aquí se comienzan a barrer las líneas con información de imagen. Estas líneas precisan que las componentes de color vayan cambiando en perfecta sincronía, para de esta forma construir la imagen en pantalla. El impulso de sincronismo horizontal y la información

de color se generan mediante el mismo método de temporización descrito anteriormente.

6) Última línea:
   a) Se barre la última línea a nivel de negro.
   b) Se comprueba el campo:

      b1) Si está en el campo impar, se actualiza el bit señalizador de campo para indicar que a continuación se va a barrer el campo par. En el campo par solo hay 4 impulsos de igualación anterior, por lo que antes de comenzar su barrido también se actualiza el contador asociado.

      b2) Si está en el campo par, se actualiza el bit señalizador de campo para indicar que a continuación se va a barrer el campo impar. En el campo impar vuelve a haber 5 impulsos de igualación anterior, por lo que es necesario que el contador asociado se actualice antes. Por último, se ha de añadir media línea más a nivel de negro, para de esta forma cumplir con la especificación de 312.5 líneas por campo.

7) Chequea tecla: Justo al final del campo par se comprueba si la tecla de cambio se ha pulsado. Si no es así se continua con el barrido, pero si por el contrario se ha pulsado, se interrumpe el barrido y se cambia al siguiente modo de funcionamiento. Tras la pulsación de la tecla, se espera unos 20ms para eliminar los rebotes de contacto, y luego se espera a que se suelte la tecla. De esta forma se evitan saltos inesperados a otros modos de funcionamiento. Por último, se comprueba el modo de funcionamiento a través del señalizador de modo, y se produce el salto al siguiente modo.

Seguidamente, en la figura 5.2 se muestra el diagrama de funciones para la secuencia del programa dentro de cada modo de barrido.

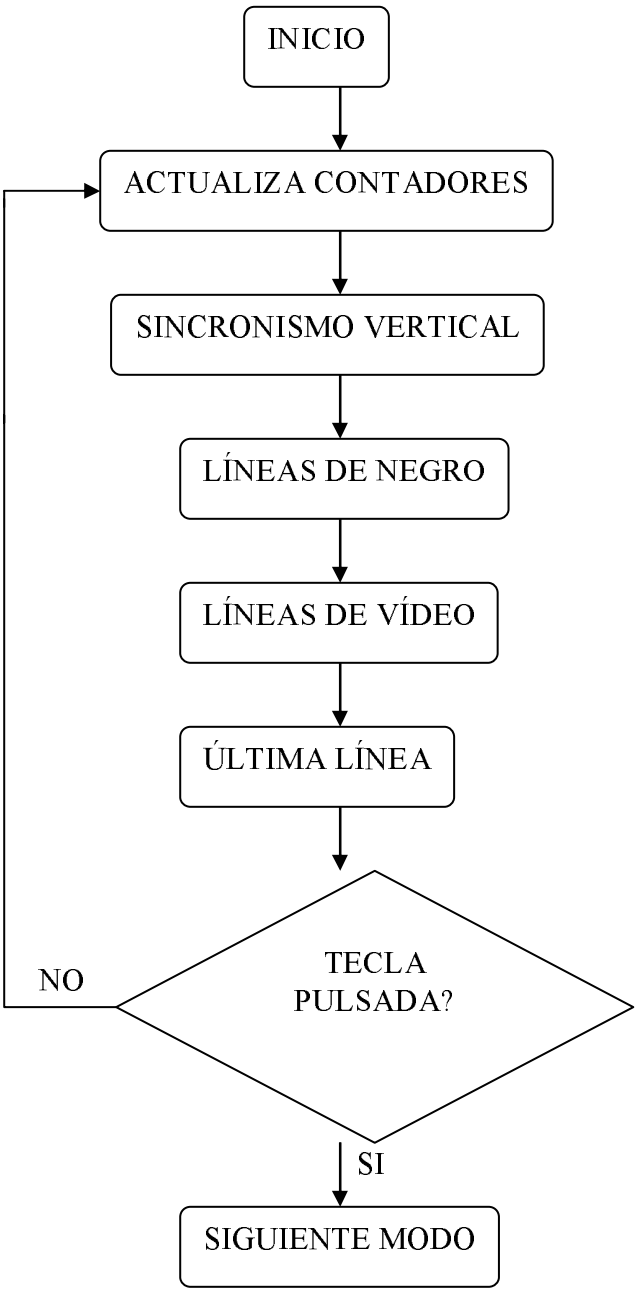


Figura 5.2 – Secuencia del software dentro de cada modo de funcionamiento.

5.1.2. CÓDIGO FUENTE.

A continuación se muestra el código fuente del generador de vídeo. Se encuentra escrito en lenguaje ensamblador por el motivo explicado anteriormente, y en el se puede ver con detalle todo lo que se ha expuesto. Para garantizar su comprensión se comenta línea a línea.

```
;Generador de Vídeo para Televisión.
;Versión: 1.0.
;Autor: José Juan Bautista Pulido.
;Director: Andrés Roldán Aranda.
;Programa para PIC16F84/10.
;Genera patrones de video a color:
;Barras de color, Raster(W,R,G,B), Damero, y Rejilla.
;El video es entrelazado.
;Portb(0) -> señal de sincronismo compuesto(SYNC).
;Portb(2-4) -> señales componentes de color(R,G,B).
;Porta(2) -> pulsador para cambio de modo -> conectar resistencia de pull-up de 20k.

;IGUALDADES
w       equ  0      ;destino w
f       equ  1      ;destino registro
status  equ  0x03   ;registro estado
porta   equ  0x05   ;puerta a
portb   equ  0x06   ;puerta b
trisa   equ  0x85   ;registro triestado puerta a
trisb   equ  0x86   ;registro triestado puerta b
c1      equ  0x0c   ;contador prop. general 1
c2      equ  0x0d   ;contador prop. general 2
ciia    equ  0x0e   ;contador iia
cisv    equ  0x0f   ;contador isv
ciip    equ  0x10   ;contador iip
flag    equ  0x11   ;registro banderas
cbl     equ  0x12   ;contador lineas en negro
cvl1    equ  0x13   ;contador lineas de video 1
cvl2    equ  0x14   ;contador lineas de video 2
cvl3    equ  0x15   ;contador lineas de video 3
cvl4    equ  0x16   ;contador lineas de video 4
rp0     equ  5      ;selección pagina memoria
sync    equ  0      ;bit 0 portb(sincronismo compuesto)
field   equ  0      ;bit 0 registro banderas(bit de campo)
```

```
cb          equ   1           ;bit 1 registro banderas(modo barras de color)
ra1         equ   2           ;bit 2 registro banderas(modo raster1)
ra2         equ   3           ;bit 3 registro banderas(modo raster2)
ra3         equ   4           ;bit 4 registro banderas(modo raster3)
ra4         equ   5           ;bit 5 registro banderas(modo raster4)
dam         equ   6           ;bit 6 registro banderas(modo damero)
rej         equ   7           ;bit 7 registro banderas(modo rejilla)
blanco      equ   b'00011101' ;componentes para blanco
amarillo    equ   b'00011001' ;componentes para amarillo
cian        equ   b'00010101' ;componentes para cian
verde       equ   b'00010001' ;componentes para verde
magenta     equ   b'00001101' ;componentes para magenta
rojo        equ   b'00001001' ;componentes para rojo
azul        equ   b'00000101' ;componentes para azul
negro       equ   b'00000001' ;componentes para negro


;RESET
            org     0x00        ;vector de reset
            goto    inicio      ;comienza el programa


;INICIO
inicio      bsf     status,rp0  ;selecciona página 1
            movlw   b'00011111' ;puerta a:
            movwf   trisa       ;todo entradas
            movlw   b'00000000' ;puerta b:
            movwf   trisb       ;todo salidas
            bcf     status,rp0  ;vuelve a página 0
            movlw   negro       ;sincronismo a nivel alto y
            movwf   portb       ;componentes para nivel de negro
            goto    inicolbar   ;comienza con barras de color


;SINCRONISMO VERTICAL
iia         bcf     portb,sync  ;impulsos de igualación anterior
            nop                 ;sync -> 2.35us a nivel bajo(se consiguen 2.4us)
            nop                 ;se introducen nop
            nop                 ;hasta conseguir la temporización
            nop                 ;
            nop                 ;
            bsf     portb,sync  ;sync -> 29.65us a nivel alto(se consiguen 29.6us)
            movlw   d'23'       ;se introduce bucle de retardo
            movwf   c1          ;hasta conseguir la temporización
loop1       decfsz  c1,f        ;
            goto    loop1       ;68 instr
            decfsz  ciia,f      ;
            goto    iia         ;
```

```
            nop                         ;
isv         bcf         portb,sync      ;impulsos de sincronismo vertical
            movlw       d'22'           ;sync -> 27.3us a nivel bajo(se consiguen 27.2us)
            movwf       c1              ;
loop2       decfsz      c1,f            ;
            goto        loop2           ;65 instr
            bsf         portb,sync      ;sync -> 4.7us a nivel alto(se consiguen 4.8us)
            movlw       d'2'            ;
            movwf       c1              ;
loop3       decfsz      c1,f            ;
            goto        loop3           ;5 instr
            nop                         ;
            decfsz      cisv,f          ;
            goto        isv             ;
            nop                         ;
iip         bcf         portb,sync      ;impulsos de igualación posterior
            nop                         ;sync -> 2.35us a nivel bajo(se consiguen 2.4us)
            nop                         ;
            nop                         ;
            nop                         ;
            nop                         ;
            bsf         portb,sync      ;sync -> 29.65us a nivel alto(se consiguen 29.6us)
            movlw       d'23'           ;
            movwf       c1              ;
loop4       decfsz      c1,f            ;
            goto        loop4           ;68 instr
            decfsz      ciip,f          ;
            goto        iip             ;
            nop                         ;


;LINEAS DE NEGRO
blklin      bcf         portb,sync      ;impulso de sincronismo horizontal
            movlw       d'3'            ;ish -> 4.7us(se consiguen 4.8us)
            movwf       c1              ;
loop5       decfsz      c1,f            ;
            goto        loop5           ;8 instr
            nop                         ;
            bsf         portb,sync      ;portico posterior +
            movlw       d'47'           ;video en negro +
            movwf       c1              ;portico anterior
loop6       decfsz      c1,f            ;total -> 59.3us(se consiguen 59.2us)
            goto        loop6           ;140 instr
            nop                         ;
            nop                         ;
            decfsz      cbl,f           ;
            goto        blklin          ;
            return                      ;retorna
```

```
;ULTIMA LINEA
ultlin      nop                     ;
            nop                     ;
ultlin2     movlw       d'2'        ;
            movwf       c1          ;
loop7       decfsz      c1,f        ;
            goto        loop7       ;5 instr
            bsf         portb,sync  ;pph + video + pah
            btfss       flag,field  ;lee campo
            goto        cpar        ;si campo par -> salta a cpar
cimpar      bcf         flag,field  ;si campo impar
            movlw       d'4'        ;cambia bandera a campo par,
            movwf       ciia        ;y actualiza con 4 el contador
            movlw       d'41'       ;de impulsos de igualación anterior
            movwf       c1          ;retardo hasta conseguir
loop8       decfsz      c1,f        ;la temporización
            goto        loop8       ;122 instr
            return                  ;retorna
cpar        bsf         flag,field  ;si campo par
            movlw       d'5'        ;cambia bandera a campo impar,
            movwf       ciia        ;y actualiza con 5 el contador
            movlw       d'46'       ;de impulsos de igualación anterior
            movwf       c1          ;retardo hasta
loop9       decfsz      c1,f        ;terminar la linea
            goto        loop9       ;137 instr
            nop                     ;
            nop                     ;
            bcf         portb,sync  ;media linea más en el campo par
            movlw       d'3'        ;ish
            movwf       c1          ;
loop10      decfsz      c1,f        ;
            goto        loop10      ;8 instr
            nop                     ;
            bsf         portb,sync  ;pph + video + pah
            movlw       d'15'       ;retardo hasta conseguir
            movwf       c1          ;la temporización
loop11      decfsz      c1,f        ;
            goto        loop11      ;44 instr
            nop                     ;
            nop                     ;
leetec      btfsc       porta,2     ;lee bit 2 porta -> si es 0 salta
            return                  ;retorna
            clrf        c1          ;
            movlw       d'65'       ;
            movwf       c2          ;
```

```
delay       decfsz      c1,f        ;retardo
            goto        delay       ;de 20ms aprox.
            decfsz      c2,f        ;para eliminar los
            goto        delay       ;rebotes de contacto
leetec2     btfss       porta,2     ;lee bit 2 porta -> si es 1 salta
            goto        leetec2     ;                -> si es 0 espera
actmod      btfsc       flag,cb     ;si modo 1 -> modo 2
            goto        iniraster1  ;salta a raster1
            btfsc       flag,ra1    ;si modo 2 -> modo 3
            goto        iniraster2  ;salta a raster2
            btfsc       flag,ra2    ;si modo 3 -> modo 4
            goto        iniraster3  ;salta a raster3
            btfsc       flag,ra3    ;si modo 4 -> modo 5
            goto        iniraster4  ;salta a raster4
            btfsc       flag,ra4    ;si modo 5 -> modo 6
            goto        inidamero   ;salta a damero
            btfsc       flag,dam    ;si modo 6 -> modo 7
            goto        inirejilla  ;salta a rejilla
            btfsc       flag,rej    ;si modo 7 -> modo 1
            goto        inicolbar   ;salta a colbar

;BARRAS DE COLOR
inicolbar   bcf         flag,dam    ;actualiza bits
            bsf         flag,cb     ;de modo
            bsf         flag,field  ;comienza con campo impar
            movlw       d'5'        ;carga 5 en el contador
            movwf       ciia        ;de impulsos de igualación anterior
colbar      movlw       d'5'        ;carga 5 en el contador
            movwf       cisv        ;de impulsos de sincronismo vertical,
            movwf       ciip        ;y de impulsos de igualación posterior
            movlw       d'16'       ;carga 16 en el contador
            movwf       cbl         ;de lineas de negro
            movlw       d'255'      ;carga 255 en el
            movwf       cvl1        ;primer contador de lineas de video
            movlw       d'255'      ;carga 255 en el
            movwf       cvl2        ;segundo contador de lineas de video
            nop                     ;
            nop                     ;
            nop                     ;
            call        iia         ;llama a sincronismo vertical
vidlin      bcf         portb,sync  ;lineas de video
            movlw       d'3'    ;impulso de sincronismo horizontal
            movwf       c1          ;ish -> 4.7us(se consiguen 4.8us)
```

```
loop12      decfsz      c1,f        ;
            goto        loop12      ;8 instr
            nop                     ;
            bsf         portb,sync  ;portico posterior
            movlw       d'3'        ;pph -> 5.8us(se consiguen 5.6us)
            movwf       c1          ;
loop13      decfsz      c1,f        ;
            goto        loop13      ;8 instr
            nop                     ;
            nop                     ;
            movlw       blanco      ;
            movwf       portb       ;barra blanca
            movlw       d'4'        ;duración 6.4us
            movwf       c1          ;
loop14      decfsz      c1,f        ;
            goto        loop14      ;11 instr
            nop                     ;
            movlw       amarillo    ;
            movwf       portb       ;barra amarilla
            movlw       d'4'        ;duración 6.4us
            movwf       c1          ;
loop15      decfsz      c1,f        ;
            goto        loop15      ;11 instr
            nop                     ;
            movlw       cian        ;
            movwf       portb       ;barra cian
            movlw       d'4'        ;duración 6.4us
            movwf       c1          ;
loop16      decfsz      c1,f        ;
            goto        loop16      ;11 instr
            nop                     ;
            movlw       verde       ;
            movwf       portb       ;barra verde
            movlw       d'4'        ;duración 6.4us
            movwf       c1          ;
loop17      decfsz      c1,f        ;
            goto        loop17      ;11 instr
            nop                     ;
            movlw       magenta     ;
            movwf       portb       ;barra magenta
            movlw       d'4'        ;duración 6.4us
            movwf       c1          ;
loop18      decfsz      c1,f        ;
            goto        loop18      ;11 instr
            nop                     ;
```

```
movlw   rojo                ;
        movwf   portb       ;barra roja
        movlw   d'4'        ;duración 6.4us
        movwf   c1          ;
loop19  decfsz  c1,f        ;
        goto    loop19      ;11 instr
        nop                 ;
        movlw   azul        ;
        movwf   portb       ;barra azul
        movlw   d'4'        ;duración 6.4us
        movwf   c1          ;
loop20  decfsz  c1,f        ;
        goto    loop20      ;11 instr
        nop                 ;
        movlw   negro       ;
        movwf   portb       ;barra negra
        movlw   d'33'       ;duración 6.4us
        decfsz  cvl2,f      ;decrementa linea:
        goto    nosuma      ;si linea < 255 -> continua
suma    addwf   cvl1,f      ;si linea = 255 -> suma 33 lineas más
nosuma  nop                 ;en total 288 lineas
        movlw   d'4'        ;
        movwf   c1          ;también se incluye tiempo
loop21  decfsz  c1,f        ;para portico anterior
        goto    loop21      ;11 instr
        decfsz  cvl1,f      ;
        goto    vidlin      ;
        nop                 ;
        bcf     portb,sync  ;ish ultima linea
        call    ultlin      ;ultima linea para cargar variables
        goto    colbar      ;actualiza contadores para un nuevo barrido


;RASTER1
iniraster1  bcf     flag,cb     ;actualiza bits
            bsf     flag,ra1    ;de modo
            bsf     flag,field  ;comienza con campo impar
            movlw   d'5'        ;carga 5 en el contador
            movwf   ciia        ;de impulsos de igualación anterior
raster1     movlw   d'5'        ;carga 5 en el contador
            movwf   cisv        ;de impulsos de sincronismo vertical,
            movwf   ciip        ;y de impulsos de igualación posterior
            movlw   d'16'       ;carga 16 en el contador
            movwf   cbl         ;de lineas de negro
            movlw   d'255'      ;carga 255 en el
            movwf   cvl1        ;primer contador de lineas de video
            movlw   d'255'      ;carga 255 en el
            movwf   cvl2        ;segundo contador de lineas de video
```

```
            nop                         ;
            nop                         ;
            nop                         ;
            call        iia             ;llama a sincronismo vertical
vidlin1     bcf         portb,sync      ;lineas de video
            movlw       d'3'            ;impulso de sincronismo horizontal
            movwf       c1              ;ish -> 4.7us(se consiguen 4.8us)
loop22      decfsz      c1,f            ;
            goto        loop22          ;8 instr
            nop                         ;
            bsf         portb,sync      ;portico posterior
            movlw       d'3'            ;pph -> 5.8us(se consiguen 5.6us)
            movwf       c1              ;
loop23      decfsz      c1,f            ;
            goto        loop23          ;8 instr
            nop                         ;
            nop                         ;
            movlw       blanco          ;
            movwf       portb           ;blanco
            movlw       d'41'           ;durante toda la linea
            movwf       c1              ;
loop24      decfsz      c1,f            ;
            goto        loop24          ;122 instr
            movlw       d'33'           ;
            decfsz      cvl2,f          ;decrementa linea:
            goto        nosuma1         ;si linea < 255 -> continua
suma1       addwf       cvl1,f          ;si linea = 255 -> suma 33 lineas más
nosuma1     movlw       negro           ;
            movwf       portb           ;barra negra
            decfsz      cvl1,f          ;en total 288 lineas
            goto        vidlin1         ;también se incluye tiempo
            nop                         ;para portico anterior
            bcf         portb,sync      ;ish ultima linea
            call        ultlin          ;ultima linea para cargar variables
            goto        raster1         ;actualiza contadores para un nuevo barrido


;RASTER2
iniraster2  bcf         flag,ra1        ;actualiza bits
            bsf         flag,ra2        ;de modo
            bsf         flag,field      ;comienza con campo impar
            movlw       d'5'            ;carga 5 en el contador
            movwf       ciia            ;de impulsos de igualación anterior
raster2     movlw       d'5'            ;carga 5 en el contador
            movwf       cisv            ;de impulsos de sincronismo vertical,
            movwf       ciip            ;y de impulsos de igualación posterior
            movlw       d'16'           ;carga 16 en el contador
            movwf       cbl             ;de lineas de negro
```

```
            movlw   d'255'      ;carga 255 en el
            movwf   cvl1        ;primer contador de lineas de video
            movlw   d'255'      ;carga 255 en el
            movwf   cvl2        ;segundo contador de lineas de video
            nop                 ;
            nop                 ;
            nop                 ;
            call    iia         ;llama a sincronismo vertical
vidlin2     bcf     portb,sync  ;lineas de video
            movlw   d'3'        ;impulso de sincronismo horizontal
            movwf   c1          ;ish -> 4.7us(se consiguen 4.8us)
loop25      decfsz  c1,f        ;
            goto    loop25      ;8 instr
            nop                 ;
            bsf     portb,sync  ;portico posterior
            movlw   d'3'        ;pph -> 5.8us(se consiguen 5.6us)
            movwf   c1          ;
loop26      decfsz  c1,f        ;
            goto    loop26      ;8 instr
            nop                 ;
            nop                 ;
            movlw   rojo        ;
            movwf   portb       ;rojo
            movlw   d'41'       ;durante toda la linea
            movwf   c1          ;
loop27      decfsz  c1,f        ;
            goto    loop27      ;122 instr
            movlw   d'33'       ;
            decfsz  cvl2,f      ;decrementa linea:
            goto    nosuma2     ;si linea < 255 -> continua
suma2       addwf   cvl1,f      ;si linea = 255 -> suma 33 lineas más
nosuma2     movlw   negro       ;
            movwf   portb       ;barra negra
            decfsz  cvl1,f      ;en total 288 lineas
            goto    vidlin2     ;también se incluye tiempo
            nop                 ;para portico anterior
            bcf     portb,sync  ;ish ultima linea
            call    ultlin      ;ultima linea para cargar variables
            goto    raster2     ;actualiza contadores para un nuevo barrido


;RASTER3
iniraster3  bcf     flag,ra2    ;actualiza bits
            bsf     flag,ra3    ;de modo
            bsf     flag,field  ;comienza con campo impar
            movlw   d'5'        ;carga 5 en el contador
            movwf   ciia        ;de impulsos de igualación anterior
```

```
raster3     movlw   d'5'        ;carga 5 en el contador
            movwf   cisv        ;de impulsos de sincronismo vertical,
            movwf   ciip        ;y de impulsos de igualación posterior
            movlw   d'16'       ;carga 16 en el contador
            movwf   cbl         ;de lineas de negro
            movlw   d'255'      ;carga 255 en el
            movwf   cvl1        ;primer contador de lineas de video
            movlw   d'255'      ;carga 255 en el
            movwf   cvl2        ;segundo contador de lineas de video
            nop                 ;
            nop                 ;
            nop                 ;
            call    iia         ;llama a sincronismo vertical
vidlin3     bcf     portb,sync  ;lineas de video
            movlw   d'3'        ;impulso de sincronismo horizontal
            movwf   c1          ;ish -> 4.7us(se consiguen 4.8us)
loop28      decfsz  c1,f        ;
            goto    loop28      ;8 instr
            nop                 ;
            bsf     portb,sync  ;portico posterior
            movlw   d'3'        ;pph -> 5.8us(se consiguen 5.6us)
            movwf   c1          ;
loop29      decfsz  c1,f        ;
            goto    loop29      ;8 instr
            nop                 ;
            nop                 ;
            movlw   verde       ;
            movwf   portb       ;verde
            movlw   d'41'       ;durante toda la linea
            movwf   c1          ;
loop30      decfsz  c1,f        ;
            goto    loop30      ;122 instr
            movlw   d'33'       ;
            decfsz  cvl2,f      ;decrementa linea:
            goto    nosuma3     ;si linea < 255 -> continua
suma3       addwf   cvl1,f      ;si linea = 255 -> suma 33 lineas más
nosuma3     movlw   negro       ;
            movwf   portb       ;barra negra
            decfsz  cvl1,f      ;en total 288 lineas
            goto    vidlin3     ;también se incluye tiempo
            nop                 ;para portico anterior
            bcf     portb,sync  ;ish ultima linea
            call    ultlin      ;ultima linea para cargar variables
            goto    raster3     ;actualiza contadores para un nuevo barrido
```

```
;RASTER4
iniraster4      bcf         flag,ra3        ;actualiza bits
                bsf         flag,ra4        ;de modo
                bsf         flag,field      ;comienza con campo impar
                movlw       d'5'            ;carga 5 en el contador
                movwf       ciia            ;de impulsos de igualación anterior
raster4         movlw       d'5'            ;carga 5 en el contador
                movwf       cisv            ;de impulsos de sincronismo vertical,
                movwf       ciip            ;y de impulsos de igualación posterior
                movlw       d'16'           ;carga 16 en el contador
                movwf       cbl             ;de lineas de negro
                movlw       d'255'          ;carga 255 en el
                movwf       cvl1            ;primer contador de lineas de video
                movlw       d'255'          ;carga 255 en el
                movwf       cvl2            ;segundo contador de lineas de video
                nop                         ;
                nop                         ;
                nop                         ;
                call        iia             ;llama a sincronismo vertical
vidlin4         bcf         portb,sync      ;lineas de video
                movlw       d'3'            ;impulso de sincronismo horizontal
                movwf       c1              ;ish -> 4.7us(se consiguen 4.8us)
loop31          decfsz      c1,f            ;
                goto        loop31          ;8 instr
                nop                         ;
                bsf         portb,sync      ;portico posterior
                movlw       d'3'            ;pph -> 5.8us(se consiguen 5.6us)
                movwf       c1              ;
loop32          decfsz      c1,f            ;
                goto        loop32          ;8 instr
                nop                         ;
                nop                         ;
                movlw       azul            ;
                movwf       portb           ;azul
                movlw       d'41'           ;durante toda la linea
                movwf       c1              ;
loop33          decfsz      c1,f            ;
                goto        loop33          ;122 instr
                movlw       d'33'           ;
                decfsz      cvl2,f          ;decrementa linea:
                goto        nosuma4         ;si linea < 255 -> continua
suma4           addwf       cvl1,f          ;si linea = 255 -> suma 33 lineas más
nosuma4         movlw       negro           ;
                movwf       portb           ;barra negra
                decfsz      cvl1,f          ;en total 288 lineas
                goto        vidlin4         ;también se incluye tiempo
                nop                         ;para portico anterior
                bcf         portb,sync      ;ish ultima linea
```

```
            call    ultlin      ;ultima linea para cargar variables
            goto    raster4     ;actualiza contadores para un nuevo barrido


;DAMERO
inidamero   bcf     flag,ra4    ;actualiza bits
            bsf     flag,dam    ;de modo
            bsf     flag,field  ;comienza con campo impar
            movlw   d'5'        ;carga 5 en el contador
            movwf   ciia        ;de impulsos de igualación anterior
damero      movlw   d'5'        ;carga 5 en el contador
            movwf   cisv        ;de impulsos de sincronismo vertical,
            movwf   ciip        ;y de impulsos de igualación posterior
            movlw   d'16'       ;carga 16 en el contador
            movwf   cbl         ;de lineas de negro
            movlw   d'18'       ;carga 18 en el
            movwf   cvl1        ;primer contador de lineas de video
            movlw   d'18'       ;carga 18 en el
            movwf   cvl2        ;segundo contador de lineas de video
            movlw   d'8'        ;carga 8 en el
            movwf   cvl3        ;tercer contador de lineas de video,
            nop                 ;
            call    iia         ;llama a sincronismo vertical
vidlin5     bcf     portb,sync  ;lineas de video
            nop                 ;impulso de sincronismo horizontal
            nop                 ;ish -> 4.7us(se consiguen 4.8us)
            nop                 ;
vidlin7     movlw   d'2'        ;
            movwf   c1          ;
loop34      decfsz  c1,f        ;
            goto    loop34      ;5 instr
            nop                 ;
            bsf     portb,sync  ;portico posterior
            movlw   d'3'        ;pph -> 5.8us(se consiguen 5.6us)
            movwf   c1          ;
loop35      decfsz  c1,f        ;
            goto    loop35      ;8 instr
            nop                 ;
            nop                 ;
            movlw   negro       ;
            movwf   portb       ;barra negra
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
```

```
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
```

```
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
```

```
            movlw   blanco      ;
            movwf   portb       ;barra blanca
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            movlw   negro       ;
            movwf   portb       ;barra negra
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            movlw   blanco      ;
            movwf   portb       ;barra blanca
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            nop                 ;
            movlw   negro       ;barra negra
            movwf   portb       ;también se incluye tiempo
            nop                 ;para portico anterior
            movlw   d'18'       ;carga w con 18,
            decfsz  cvl1,f      ;
            goto    vidlin5     ;
            movwf   cvl1        ;y al salir actualiza cvl1
vidlin6     bcf     portb,sync  ;lineas de video
            movlw   d'3'        ;impulso de sincronismo horizontal
            movwf   c1          ;ish -> 4.7us(se consiguen 4.8us)
loop36      decfsz  c1,f        ;
            goto    loop36      ;8 instr
            nop                 ;
            bsf     portb,sync  ;portico posterior
            movlw   d'3'        ;pph -> 5.8us(se consiguen 5.6us)
            movwf   c1          ;
loop37      decfsz  c1,f        ;
            goto    loop37      ;8 instr
            nop                 ;
            nop                 ;
            movlw   blanco      ;
            movwf   portb       ;barra blanca
            nop                 ;
            nop                 ;
```

```
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
```

```
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   negro   ;
movwf   portb   ;barra negra
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
nop             ;
movlw   blanco  ;
movwf   portb   ;barra blanca
nop             ;
nop             ;
```

```
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            movlw       negro       ;
            movwf       portb       ;barra negra
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            movlw       blanco      ;
            movwf       portb       ;barra blanca
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            movlw       negro       ;
            movwf       portb       ;barra negra
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            nop                     ;
            movlw       blanco      ;
            movwf       portb       ;barra blanca
            movlw       negro       ;también se incluye tiempo
            movwf       portb       ;para portico anterior
            movlw       d'18'       ;carga w con 18,
            decfsz      cvl2,f      ;
            goto        vidlin6     ;
            movwf       cvl2        ;y al salir actualiza cvl2
            bcf         portb,sync  ;ish
            decfsz      cvl3,f      ;decrementa cvl3,
            goto        vidlin7     ;y se repite el bucle 8 veces
            call        ultlin2     ;ultima linea para cargar variables
            goto        damero      ;actualiza contadores para un nuevo barrido


;REJILLA
inirejilla  bcf         flag,dam    ;actualiza bits
            bsf         flag,rej    ;de modo
            bsf         flag,field  ;comienza con campo impar
            movlw       d'5'        ;carga 5 en el contador
            movwf       ciia        ;de impulsos de igualación anterior
```

```
rejilla     movlw   d'5'        ;carga 5 en el contador
            movwf   cisv        ;de impulsos de sincronismo vertical,
            movwf   ciip        ;y de impulsos de igualación posterior
            movlw   d'16'       ;carga 16 en el contador
            movwf   cbl         ;de lineas de negro
            movlw   d'3'        ;carga 3 en el
            movwf   cvl1        ;primer contador de lineas de video
            movlw   d'33'       ;carga 33 en el
            movwf   cvl2        ;segundo contador de lineas de video
            movlw   d'8'        ;carga 8 en el
            movwf   cvl3        ;tercer contador de lineas de video,
            nop                 ;
            call    iia         ;llama a sincronismo vertical
vidlin8     bcf     portb,sync  ;lineas de video
            nop                 ;impulso de sincronismo horizontal
            nop                 ;ish -> 4.7us(se consiguen 4.8us)
            nop                 ;
vidlin10    movlw   d'2'        ;
            movwf   c1          ;
loop38      decfsz  c1,f        ;
            goto    loop38      ;5 instr
            nop                 ;
            bsf     portb,sync  ;portico posterior
            movlw   d'3'        ;pph -> 5.8us(se consiguen 5.6us)
            movwf   c1          ;
loop39      decfsz  c1,f        ;
            goto    loop39      ;8 instr
            nop                 ;
            nop                 ;
            movlw   blanco      ;
            movwf   portb       ;blanco
            movlw   d'42'       ;durante toda la linea
            movwf   c1          ;
loop40      decfsz  c1,f        ;
            goto    loop40      ;125 instr
            movlw   negro       ;también se incluye tiempo
            movwf   portb       ;para portico anterior
            movlw   d'3'        ;carga w con 3,
            decfsz  cvl1,f      ;
            goto    vidlin8     ;
            movwf   cvl1        ;y al salir actualiza cvl1
vidlin9     bcf     portb,sync  ;lineas de video
            movlw   d'3'        ;impulso de sincronismo horizontal
            movwf   c1          ;ish -> 4.7us(se consiguen 4.8us)
loop41      decfsz  c1,f        ;
            goto    loop41      ;8 instr
            nop                 ;
            bsf     portb,sync  ;portico posterior
```

```
        movlw   d'3'            ;pph -> 5.8us(se consiguen 5.6us)
        movwf   c1              ;
loop42  decfsz  c1,f            ;
        goto    loop42          ;8 instr
        nop                     ;
        nop                     ;
        movlw   b'00011100'     ;
        addwf   portb,f         ;barra blanca
        subwf   portb,f         ;
        nop                     ;
        nop                     ;
        nop                     ;
        movlw   d'3'            ;
        movwf   c1              ;
loop43  decfsz  c1,f            ;
        goto    loop43          ;8 instr
        movlw   b'00011100'     ;
        addwf   portb,f         ;barra blanca
        subwf   portb,f         ;
        nop                     ;
        nop                     ;
        nop                     ;
        movlw   d'3'            ;
        movwf   c1              ;
loop44  decfsz  c1,f            ;
        goto    loop44          ;8 instr
        movlw   b'00011100'     ;
        addwf   portb,f         ;barra blanca
        subwf   portb,f         ;
        nop                     ;
        nop                     ;
        nop                     ;
        movlw   d'3'            ;
        movwf   c1              ;
loop45  decfsz  c1,f            ;
        goto    loop45          ;8 instr
        movlw   b'00011100'     ;
        addwf   portb,f         ;barra blanca
        subwf   portb,f         ;
        nop                     ;
        nop                     ;
        nop                     ;
        movlw   d'3'            ;
        movwf   c1              ;
loop46  decfsz  c1,f            ;
        goto    loop46          ;8 instr
        movlw   b'00011100'     ;
        addwf   portb,f         ;barra blanca
```

```
            subwf       portb,f         ;
            nop                         ;
            nop                         ;
            nop                         ;
            movlw       d'3'            ;
            movwf       c1              ;
loop47      decfsz      c1,f            ;
            goto        loop47          ;8 instr
            movlw       b'00011100'     ;
            addwf       portb,f         ;barra blanca
            subwf       portb,f         ;
            nop                         ;
            nop                         ;
            nop                         ;
            movlw       d'3'            ;
            movwf       c1              ;
loop48      decfsz      c1,f            ;
            goto        loop48          ;8 instr
            movlw       b'00011100'     ;
            addwf       portb,f         ;barra blanca
            subwf       portb,f         ;
            nop                         ;
            nop                         ;
            nop                         ;
            movlw       d'3'            ;
            movwf       c1              ;
loop49      decfsz      c1,f            ;
            goto        loop49          ;8 instr
            movlw       b'00011100'     ;
            addwf       portb,f         ;barra blanca
            subwf       portb,f         ;
            nop                         ;
            nop                         ;
            nop                         ;
            movlw       d'2'            ;
            movwf       c1              ;
loop50      decfsz      c1,f            ;
            goto        loop50          ;5 instr
            nop                         ;
            nop                         ;
            movlw       b'00011100'     ;
            ddwf        portb,f         ;barra blanca
            subwf       portb,f         ;también se incluye tiempo
            nop                         ;para portico anterior
            movlw       d'33'           ;carga w con 33,
            decfsz      cvl2,f          ;
            goto        vidlin9         ;
            movwf       cvl2            ;y al salir actualiza cvl2
```

```
bcf         portb,sync    ;ish
decfsz      cvl3,f        ;decrementa cvl3,
goto        vidlin10      ;y se repite el bucle 8 veces
call        ultlin2       ;ultima linea para cargar variables
goto        rejilla       ;actualiza contadores para un nuevo barrido

end
```

5.2. SOFTWARE DEL MONITOR.

SOFTWARE DEL MONITOR

La gestión del monitor de vídeo también se encargo a un microcontrolador. En este caso el microcontrolador escogido para el control del sistema es el PIC16F876-04. El control de todos los módulos conectados al mismo se realiza a través de su Programa de Aplicación, esta vez realizado en lenguaje de medio nivel.

El programa se confeccionó en Lenguaje 'C'. A diferencia de la edición con lenguaje ensamblador, la edición de programas en lenguaje 'C' es mucho más rápida y compacta, pero por el contrario genera códigos más extensos. En este caso la velocidad del procesador no es un factor crítico, como sucedía en el caso anterior, pero si que se requiere más capacidad de memoria, debido sobre todo a la necesidad de almacenar multitud de mensajes y menús.

Para la creación del código fuente se utilizó la herramienta de desarrollo C2C++ de la empresa Pavel Baranov. Esta herramienta incluye:

- Un editor de texto, para la confección del código fuente.
- Un compilador, para la creación del código ensamblador.

Para ensamblar el código proporcionado por el compilador, se utiliza nuevamente la herramienta MPLAB de Microchip.

El formato empleado para almacenar el programa también es de tipo hexadecimal ".HEX ", y para cargarlo en la memoria EEPROM volvemos a utilizar el sencillo y conocido programador de microcontroladores "TE-20".

5.2.1. DIAGRAMA DE FUNCIONES.

El programa en 'C' se ha dividido en dos partes o MÓDULOS. Al ser el programa de la aplicación muy extenso, separándolo en módulos se posee un mejor control del mismo. Así, por ejemplo, hay un módulo llamado LCD.C que contiene todas las funciones necesarias para el control del display. Si por algún motivo se apreciara un error de software con relación a la representación en pantalla, se recurriría a este módulo para intervenir sobre él, no siendo necesaria ninguna intervención en el otro módulo.

A continuación se verá, de forma esquemática, el proceso que sigue el programa de la aplicación.

El programa realiza las siguientes funciones:

1) Acepta por la Puerta B las ordenes provenientes de un teclado matricial, para de esta forma poder seleccionar los diferentes modos de funcionamiento y las opciones de los mismos.
2) Utiliza la Puerta B junto con los Bits 0 a 2 de la Puerta A para el control de una pantalla LCD, donde se muestra toda la información necesaria para el funcionamiento y control del sistema.
3) Acepta por el terminal 3 del PORTA la información de campo que le suministra el separador de sincronismos.
4) Acepta por el terminal 4 del PORTA la información de sincronismo vertical (comienzo de campo) proveniente del separador de sincronismos. Tal como se comentó en la descripción del monitor, esta señal provoca una interrupción mediante la cual se recarga el contador de líneas. Cuando el contador de líneas se desborda, este produce otra interrupción la cual provoca el disparo del osciloscopio.
5) Acepta por el terminal 0 del PORTC la información de sincronismo compuesto (impulsos de línea) que suministra el separador de sincronismos. Esta señal hace avanzar el contador de líneas, llevando de esta forma la cuenta de todos los impulsos de interés.
6) Utiliza el terminal 1 del PORTC para controlar el disparo de la señal de vídeo que se pretende visualizar.

7) Utiliza el terminal 2 del PORTC para controlar un buzzer.

Y dispone de 4 modos de funcionamiento:

1) Modo vídeo.
2) Modo T.X.T.
3) Modo V.I.T.
4) Modo continuo.

El modo de operación es el siguiente:

1) Tras conectarse la alimentación, el sistema se resetea automáticamente.
2) Después del reset se inicia el sistema.
3) Tras el inicio se configuran y habilitan las interrupciones, y luego se espera a que se produzca alguna interrupción para poder atenderla.

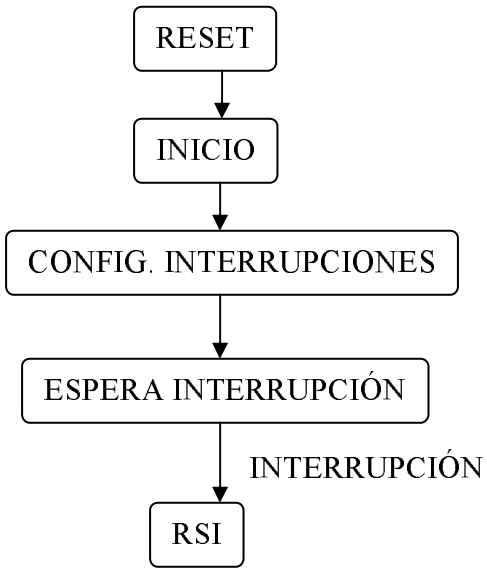


Figura 5.3 - Secuencia principal del software del monitor de vídeo.

Inicio.

Durante la secuencia de inicio, se realizan las siguientes tareas:

1) Se configuran las Puertas A y C (entradas o salidas) para las funciones descritas anteriormente.
2) Se hace sonar el buzzer dos veces para indicar que el microcontrolador a conseguido arrancar.
3) Se configura la Puerta B para el control del display, y se muestran los mensajes de inicio.
4) Se muestra el menú principal.
5) Se configura la Puerta B para el control de teclado, y se configuran los Contadores TMR0 y TMR1 para las funciones antes mencionadas.

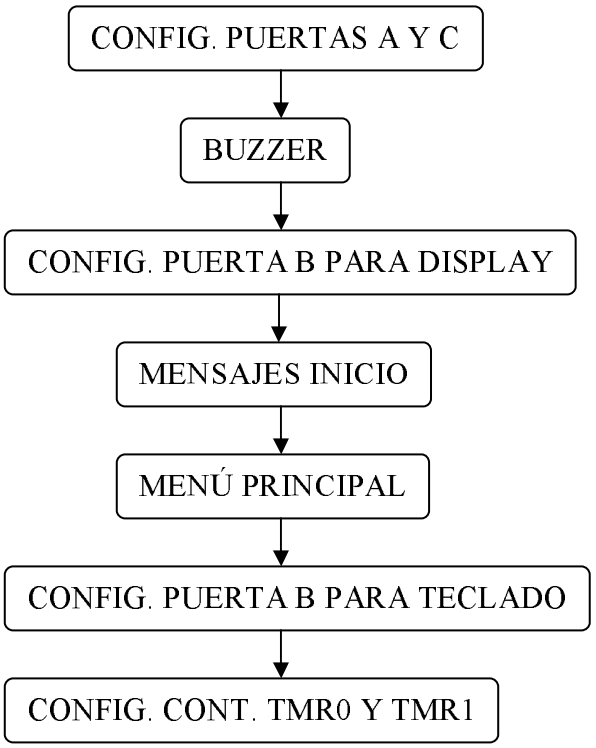


Figura 5.4 - Secuencia de inicio del software del monitor de vídeo.

Todas las tareas que se realizan durante el inicio se han comentado detalladamente en el apartado del microcontrolador. Cada vez que se ha comentado uno de los periféricos utilizados en el sistema, la explicación se ha acompañado de la configuración y modo de operación del mismo, incluyendo el software asociado.

Lo único que no se ha expuesto todavía son los mensajes de inicio y el menú de operación del sistema. Estos mensajes y menús se muestran y comentan a continuación.

Los mensajes de inicio se muestran en el mismo orden en el que van apareciendo, y son los siguientes:

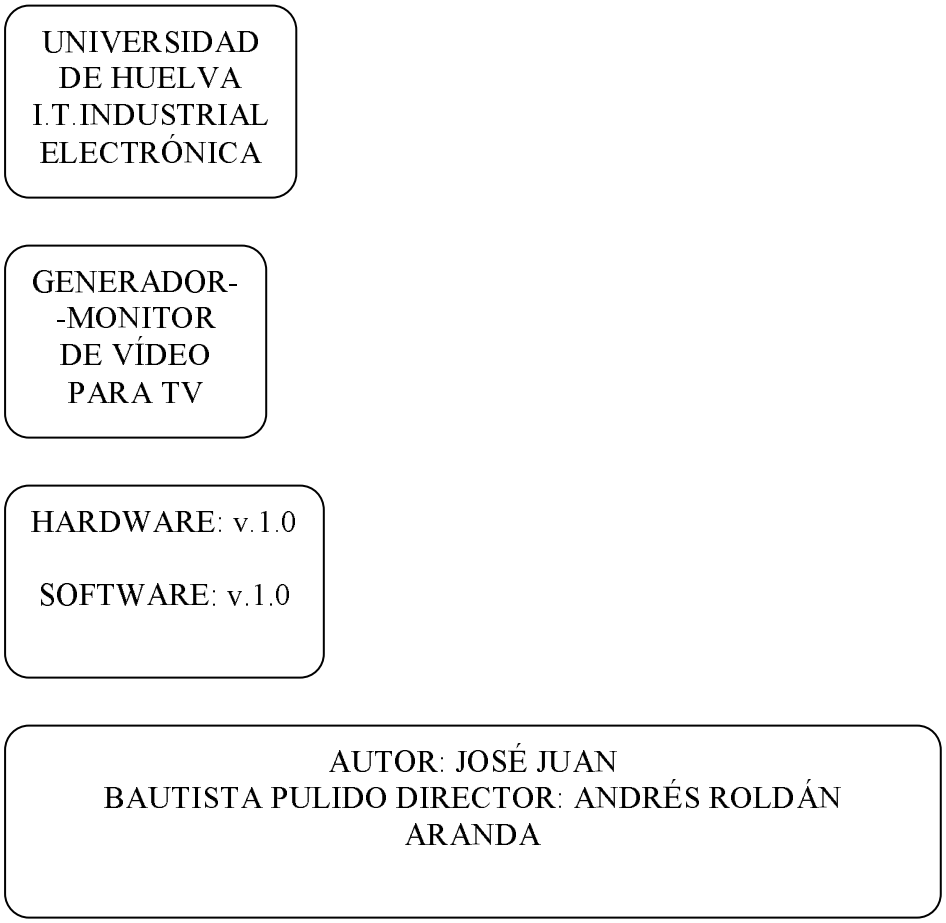
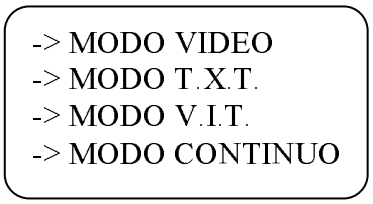


Figura 5.5 – Aspecto de los mensajes de inicio del monitor de vídeo.

El menú principal tiene el siguiente aspecto:



Tal como se explicó en el apartado del teclado matricial, las teclas irán colocadas a la altura de las filas del display. De esta forma en el menú principal cada tecla selecciona uno de los cuatro posibles modos de funcionamiento. Las flechas que aparecen en el menú indican que, pulsando la tecla asociada a la fila ,se entra en el modo de funcionamiento correspondiente.

Los menús que aparecen al ir seleccionado los diferentes modos de funcionamiento en el menú principal son los siguientes:

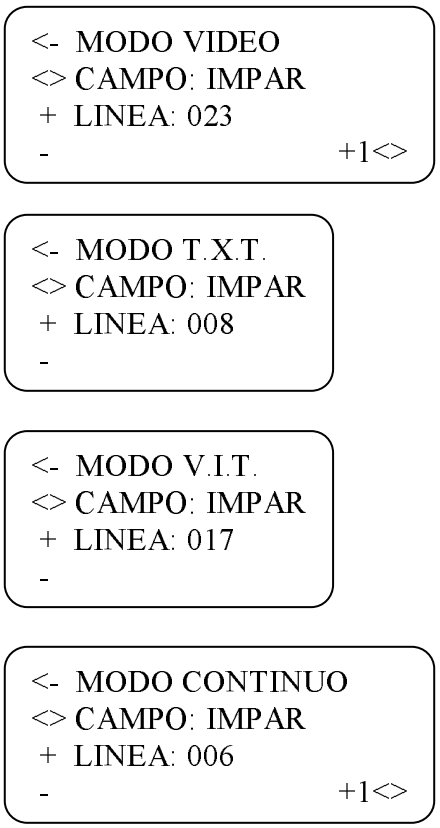


Figura 5.6 – Aspecto de los menús del monitor de vídeo.

Las flechas que aparecen en los menús anteriores indican que, pulsando la tecla asociada a la fila, se abandona el modo de funcionamiento seleccionado y se vuelve al menú principal.

El símbolo que acompaña al mensaje campo, indica la posibilidad de cambio del mismo, actualizándose el mensaje en caso de cambio.

Los símbolos + y – que acompañan al mensaje de línea, indican la posibilidad de aumentar o disminuir el número de línea, actualizándose dicho número con cada cambio.

Cada vez que se cambia de campo, se selecciona de forma automática la primera línea correspondiente al campo y modo.

Cada vez que se entra en un modo de funcionamiento, se selecciona de forma automática el campo impar (primer campo) y por tanto la primera línea del campo impar correspondiente al modo.

Por último, el primer y cuarto modo de funcionamiento también permite el cambio de líneas de diez en diez, debido al gran número de líneas existente.

Para facilitar la representación de la información en el display se diseñaron las funciones que se muestran en las figuras 5.7 y 5.8. Estas funciones hacen uso de las funciones básicas de escritura de comandos y datos, las cuales se expusieron y comentaron en el apartado del display incluido en la definición del microcontrolador.

```
void cadena(const char *cad)    //escribe cadena en pantalla
{
 char i = 0;                    //define una variable para el contaje
 while(cad[i] != 0)             //ejecuta bucle mientras no termine la cadena
 {
  lcd_write(cad[i++]);          //envía carácter de la cadena
 }
}
```

Figura 5.7 – Función que representa las cadenas en el display.

```
void menu(const char *cad1)    //escribe menu en pantalla
{
 lcd_coman(clear_lcd);         //envía comando borra pantalla y cursor a casa
 char i = 0;                   //define una variable para el contaje
 while(cad1[i] != 0)           //ejecuta bucle mientras no termine la cadena
 {
  lcd_write(cad1[i++]);        //envía carácter de la cadena
  switch(i)                    //determina posición del cursor
  {
   case 16:                    //si fin línea 1
        lcd_coman(line2);      //línea 2
        break;
   case 32:                    //si fin línea 2
        lcd_coman(line3);      //línea 3
        break;
   case 48:                    //si fin línea 3
        lcd_coman(line4);      //línea 4
        break;
  }
 }
}
```

Figura 5.7 – Función que representa los menús en el display.

La función 'cadena' utiliza la función 'lcd_write' para escribir los caracteres en la pantalla. El bucle while comprueba si se ha llegado al último carácter de la cadena antes de enviar otro carácter, para lo cual a de encontrarse el carácter nulo. Si se ha llegado al último carácter la función termina, y en el caso contrario se repite el bucle.

La función 'menú' funciona de forma parecida a la función 'cadena'. En este caso siempre se comienza borrando la pantalla, por lo que dicho comando se incluye al principio de la función. La escritura de los caracteres en la pantalla se realiza de la misma forma que en la función 'cadena', solo que en este caso se supera la capacidad de una línea. Por este motivo se comprueba cuando se alcanza el final de cada línea, y cuando esto ocurre se salta a la siguiente línea. Este control de línea es necesario porque, como ya se comento, la distribución de las líneas no es continua.

Para mostrar los menús de inicio, el menú principal y los menús de los diferentes modos de funcionamiento, basta simplemente con llamar a la función menú pasándole el texto que se quiere mostrar. Las funciones que contienen las cadenas de texto que forman los menús se mostraran en el apartado del código fuente.

Como ejemplo, en la figura 5.8 se muestra la función que representa el menú del primer modo de funcionamiento. Cada vez que se entra en un modo de funcionamiento se actualiza el modo, el campo, la línea y el salto de línea, de hay que se incluyan las instrucciones de actualización en la misma función.

```
void menu_1(void)            //menu 1
{
 menu("<- MODO VIDEO  <> CAMPO: IMPAR +  LINEA: 023  -        +1<>");
                             //actualiza menu
 modo = 1;                   //actualiza modo
 campo = 1;                  //actualiza campo
 linea = 23;                 //actualiza linea
 suma = 1;                   //actualiza sumador
 cal_tmr1();                 //calcula valor de tmr1
 buzzer();                   //buzzer al entrar en modo 1
}
```

Figura 5.8 – Función que representa el menú del primer modo.

Interrupciones.

Tal como se ha dicho, el sistema funciona mediante interrupciones.

En el apartado del microcontrolador referente a las interrupciones, se describió detalladamente el proceso que se sigue cuando se produce una interrupción. En ese apartado también se definieron todos los procedimientos necesarios para la configuración y control de las interrupciones, y como ejemplo se comentó la rutina que da servicio a las interrupciones. Esta rutina se llama RSI (Rutina servicio interrupción), y en ella se detecta quien ha producido la interrupción, y a continuación se le atiende.

También, en los aparatados del microcontrolador referentes a los contadores y a las puertas de entrada-salida, se describieron los procesos que se siguen para las diferentes interrupciones que se dan en el sistema, así como los procedimientos para la configuración de las mismas. Estas posibles interrupciones son:

- Desbordamiento de TMR1: El desbordamiento de TMR1 indica que ha llegado la línea que queremos visualizar, y por tanto se produce el disparo del osciloscopio.
- Desbordamiento de TMR0: El desbordamiento de TMR0 indica que ha comenzado el campo, y por tanto se carga el contador TMR1 para que empiece a contar las líneas.
- Cambio de Estado en la Puerta B: En cambio de estado en la puerta B indica que se ha pulsado alguna tecla del teclado matricial, y por tanto se busca la tecla responsable para realizar la función asociada.

Todos los procedimientos asociados a estas interrupciones ya se han descrito en sus correspondientes apartados, a excepción de las diferentes opciones disponibles para la interrupción del teclado según la tecla pulsada.

Las posibles opciones tras la interrupción del teclado son: cambiar de modo de funcionamiento, cambiar de campo, cambiar de línea y cambiar el salto de línea.

Cuando se solicita un cambio de modo se cambia al menú correspondiente a través de las funciones antes descritas.

Cuando se cambia de campo se realizan las siguientes tareas:

1) Se comprueba el campo.
2) Se actualiza la variable de campo con el valor del campo contrario.
3) Se comprueba el modo de funcionamiento.
4) Se actualiza la variable de línea con el valor de la primera línea del campo y modo seleccionados.
5) Se actualiza el campo en la pantalla.
6) Se actualiza la línea en la pantalla.

7) Con el nuevo valor de línea, se calcula el valor que ha de cargarse en el contador TMR1 y se carga dicho valor.
8) Se hace sonar el buzzer para indicar que se ha cambiado de campo.

A continuación, se muestra parte de la función encargada del cambio de campo. La función completa se mostrará en el apartado del código fuente debido a su gran tamaño.

```
void c_campo(void)          //cambia de campo
{
 if(campo == 1)             //si está en campo impar
 {
  campo = 0;                //pasa a campo par
  switch(modo)              //determina modo de funcionamiento
  {
   case 1:                  //si está en modo 1
        linea = 336;        //1ª linea campo par modo 1
        break;              //retorna
   case 2:                  //si está en modo 2
        linea = 321;        //1ª linea campo par modo 2
        break;              //retorna
   case 3:                  //si está en modo 3
        linea = 330;        //1ª linea campo par modo 3
        break;              //retorna
   case 4:                  //si está en modo 4
        linea = 318;        //1ª linea campo par modo 4
  }
 }
 act_campo();               //actualiza campo
 act_linea();               //actualiza linea
 cal_tmr1();                //calcula valor de tmr1
 buzzer();                  //buzzer con cambio de campo manual
}
```

Figura 5.9 – Parte de la función que controla el cambio de campo.

Cuando se cambia de línea pueden darse dos casos, que el número de línea se aumente o que se disminuya. En ambos casos las funciones utilizadas operan de forma similar, y por tanto solo se describirá una de ellas. En la figura 5.11 se muestra parte de la función que aumenta el número de línea, la cual se describe a continuación.

Cuando se aumenta la línea se dan los siguientes pasos:

1) Se incrementa el valor de la variable de línea, en una o diez unidades según el valor del salto de líneas.
2) Se comprueba el modo de funcionamiento.
3) Se comprueba el campo.
4) Dependiendo del modo y campo en que nos encontremos, se comprueba que no se haya superado la ultima línea del campo. Si se ha superado se salta a la primera línea del siguiente campo, y para ello se actualizan las variables de campo y línea.
5) Se actualiza el campo en la pantalla.
6) Se actualiza la línea en la pantalla.
7) Con el nuevo valor de línea, se calcula el valor que ha de cargarse en el contador TMR1 y se carga dicho valor.

La función 'act_campo' es la encargada de actualizar el campo en la pantalla.

Para ello sitúa el cursor en la posición adecuada y utiliza la función 'cadena' para escribir el nombre del campo.

```
void act_campo(void)        //actualiza campo en pantalla
{
 lcd_coman(0xCA);           //comando línea 2 posición 11
 if(campo == 1)             //si está en campo impar
  cadena("IMPAR");          //actualiza mensaje en pantalla
 else                       //si está en campo par
  cadena("PAR  ");          //actualiza mensaje en pantalla
}
```

Figura 5.10 – Función que actualiza el campo en la pantalla.

```
void s_linea(void)                  //aumenta número de linea
{
 linea = linea + suma;              //aumenta línea
 switch(modo)                       //determina modo de funcionamiento
 {
  case 1:                           //si está en modo 1
       if(campo == 1)               //si está en campo impar
       {
        if(linea > 310)             //si ultima linea del campo superada
        {
         campo = 0;                 //campo par
         linea = 336;               //1ª linea campo par
         buzzer();                  //buzzer con cambio de campo automático
        }
       }
       else                         //si está en campo par
       {
        if(linea > 623)             //si ultima linea del campo superada
        {
         campo = 1;                 //campo impar
         linea = 23;                //1ª linea campo impar
         buzzer();                  //buzzer con cambio de campo automático
        }
       }
       break;                       //retorna
 }
 act_campo();                       //actualiza campo
 act_linea();                       //actualiza linea
 cal_tmr1();                        //calcula valor de tmr1
}
```

Figura 5.11 – Parte de la función que controla el cambio de línea.

La función 'act_linea' actualiza la línea en la pantalla. Antes de mandar el nuevo valor, calcula los valores correspondientes a centena, decena y unidad, ya que el valor se ha de escribir carácter a carácter.

```
void act_linea(void)          //actualiza línea en pantalla
{
 char linea2;                 //define una variable para el calculo de caracteres
 centenas = linea/100;        //calcula centenas(centenas = parte entera división)
 linea2 = linea%100;          //linea2 = resto división anterior
 decenas = linea2/10;         //calcula decenas(decenas = parte entera división)
 unidades = linea2%10;        //calcula unidades(unidades = resto división anterior)
 lcd_coman(0x9A);             //comando linea 3 posición 11
 lcd_write(centenas + 48);    //escribe centenas
 lcd_write(decenas + 48);     //escribe decenas
 lcd_write(unidades + 48);    //escribe unidades
}
```

Figura 5.12 – Función que actualiza la línea en la pantalla.

La función 'cal_tmr1' toma el valor actualizado de la variable línea, y calcula el contenido de los registros alto y bajo del contador TMR1 (tmr1h y tmr1l) para poder cargar directamente el contador cuando se requiera.

```
void cal_tmr1(void)           //calcula el contenido de tmr1
{
 if(campo == 1)               //si está en campo impar
  tmr1 = 65535 - linea;       //número de impulsos para desbordamiento
 else                         //si está en campo par
  tmr1 = 65848 - linea;       //número de impulsos para desbordamiento
 tmr1b = tmr1 & 0x00FF;       //calcula lsb de tmr1
 tmr1a = (tmr1 & 0xFF00)/256; //calcula msb de tmr1
}
```

Figura 5.13 – Función que calcula el contenido del contador TMR1.

Por último, la función 'act_suma' se utiliza para cambiar el valor del salto de línea. Con la llamada a la función se cambia al otro valor disponible. La función actualiza la variable suma con el valor que corresponda (1 o 10) y actualiza el mensaje en la pantalla utilizando la función cadena.

```
void act_suma(void)                 //actualiza sumador en pantalla
{
 if((modo == 1) || (modo == 4))     //si está en modo 1 ó modo 4 -> actualiza
 {
  if(suma == 1)                     //si sumador = 1
  {
   suma = 10;                       //comando linea 4 posición 11
   cadena("+10");                   //actualiza mensaje en pantalla
  }
  else                              //si sumador = 10
  {
   suma = 1;                        //sumador = 1
   lcd_coman(0xDB);                 //comando linea 4 posición 11
   cadena(" +1");                   //actualiza mensaje en pantalla
  }
 buzzer();                          //buzzer con cambio de sumador
 }
}
```

Figura 5.14 – Función que actualiza el salto de línea.

5.2.2. CÓDIGO FUENTE.

A continuación se muestra el código fuente del monitor de vídeo. Se encuentra escrito en lenguaje 'C', y para garantizar su comprensión se divide en módulos y se incluye un comentario en cada línea.

Modulo principal.

```
//Monitor de Vídeo para Televisión.
//Versión: 1.0.
//Autor: José Juan Bautista Pulido.
//Director: Andrés Roldán Aranda.
//Programa para PIC16F876/04.
//Visualiza líneas de televisión en osciloscopio:
//Líneas de vídeo, de teletexto, y de control de calidad.


//Inclusión de librerías

#include "def.h"


//Declaración de funciones y variables

void c_campo(void);
void s_linea(void);
void b_linea(void);
void act_campo(void);
void act_linea(void);
void act_suma(void);
void cal_tmr1(void);
void buzzer(void);

int linea, tmr1;
char key, modo, campo, suma, field, tmr1a, tmr1b;
char centenas, decenas, unidades;


//Declaración de funciones y variables externas

extern void lcd_setup(void);
extern void lcd_coman(char com);
extern void lcd_write(char dat);
extern void cadena(const char *cad);
extern void inicio(void);
extern void menu_p(void);
```

```
extern void menu_1(void);
extern void menu_2(void);
extern void menu_3(void);
extern void menu_4(void);


//Definición de funciones

//Función principal
void main(void)
{
 adcon1 = 00000110b;          //configura puerta a como i/o digitales
 trisa = 00111000b;           //porta:0-2 control lcd(rs,rw,en), 3-5 video sync(field,ISV,burst)
 trisc = 00000001b;           //portc:0 csync in, 1 salida disparo, 2 salida buzzer, 3-7 salidas
 buzzer();                    //buzzer
 delay_ms(50);                //con
 buzzer();                    //alimentación
 lcd_setup();                 //configura e inicializa lcd
 inicio();                    //muestra mensajes de inicio
 menu_p();                    //muestra menu principal
 option = 00111000b;          //portb pull-ups on, tmr0 clk out flanco bajada
 t1con = 00000111b;           //tmr1:preescaler 1:1, oscilador off, clk out no sync
 trisb = 11110000b;           //configura puerta b para teclado(bits 7-4 entradas, 3-0 salidas)
 portb = 0x00;                //prepara puerta b para interrupción(bits 3-0 a 0)
 asm movf _portb,w            ;prepara puerta b para interrupción(actualiza w para comparación)
 pie1 = 00000001b;            //habilitación int. tmr1
 pir1 = 00000000b;            //bandera int. tmr1 a 0
 intcon = 11101000b;          //habilitación int. global, periféricos, tmr0, y portb, banderas a 0
 tmr0 = 0xFF;                 //actualiza tmr0 para interrupción por ISV
 bucle:                       //bucle:
 asm nop                      //sin interrupción
 goto bucle;                  //no opera
}


//RSI Desbordamiento de tmr0
void tmr0_int(void)
{
 if(modo != 0)                //si no está en menú principal
 {
  field = input_pin_port_a(3); //determina campo en la señal de video
  if(field == campo)          //si está en campo seleccionado
  {
   tmr1l = tmr1b;             //actualiza tmr1l
   tmr1h = tmr1a;             //actualiza tmr1h
  }
 }
```

```
else                        //si está en menú principal
 {
  set_bit(portc,1);         //dispara
  clear_bit(portc,1);
 }
 tmr0 = 0xFF;               //actualiza tmr0 para interrupción por ISV
}


//RSI Desbordamiento de tmr1
void tmr1_int(void)
{
 set_bit(portc,1);          //dispara,
 delay_us(115);             //y retarda
 clear_bit(portc,1);        //para visualizar linea
}


//RSI Cambio de estado en puerta b
void portb_int(void)
{
 delay_ms(15);              //retardo para eliminar rebotes de contacto
 key = portb & 0xF0;        //lee estado puerta b para determinar tecla pulsada
 switch(key)                //determina tecla pulsada
 {
  case 0xE0:                //1ª tecla pulsada
        if(modo == 0)       //si está en modo 0
         menu_1();          //va a modo 1
        else                //si está en cualquier otro modo
         menu_p();          //vuelve a modo 0
        break;              //retorna
  case 0xD0:                //2ª tecla pulsada
        if(modo == 0)       //si está en modo 0
         menu_2();          //va a modo 2
        else                //si está en cualquier otro modo
         c_campo();         //cambia de campo
        break;              //retorna
  case 0xB0:                //3ª tecla pulsada
        if(modo == 0)       //si está en modo 0
         menu_3();          //va a modo 3
        else                //si está en cualquier otro modo
         s_linea();         //sube linea
        break;              //retorna
  case 0x70:                //4ª ó 5ª tecla pulsada
        portb = 0x04;       //bit 3 a 1 para determinar pulsador
        key = portb & 0xF0; //lee estado puerta b para determinar tecla pulsada
```

```
        if(key == 0x70)        //si entrada sigue a 0 -> 4ª tecla pulsada
        {
         if(modo == 0)         //si está en modo 0
          menu_4();            //va a modo 4
         else                  //si está en cualquier otro modo
          b_linea();           //baja linea
        }
        else                   //si entrada cambia a 1 -> 5ª tecla pulsada
         act_suma();           //actualiza sumador
 }
 delay_ms(20);                 //retardo para eliminar rebotes de contacto
 trisb = 0xF0;                 //configura puerta b para teclado(bits 7-4 entradas, 3-0 salidas)
 portb = 0x00;                 //prepara puerta b para interrupción(bits 3-0 a 0)
 asm movf _portb,w             ;prepara puerta b para interrupción(actualiza w para comparación)
}


//Rutina Servicio Interrupción
void interrupt(void)
{
 clear_bit(intcon,7);          //deshabilita interrupciones
 if(intcon & 4)                //comprueba bandera interrupción tmr0(t0if)
 {
  tmr0_int();                  //RSI desbordamiento de tmr0
  clear_bit(intcon,2);         //borra bandera interrupción tmr0
  set_bit(intcon,5);           //permiso para interrupción tmr0
 }
 if(pir1 & 1)                  //comprueba bandera interrupción tmr1(tmr1if)
 {
  tmr1_int();                  //RSI desbordamiento de tmr1
  clear_bit(pir1,0);           //borra bandera interrupción tmr1
  set_bit(pie1,0);             //permiso para interrupción tmr1
 }
 if(intcon & 1)                //comprueba bandera interrupción portb(rbif)
 {
  portb_int();                 //RSI cambio de estado en puerta b
  clear_bit(intcon,0);         //borra bandera interrupción portb
  set_bit(intcon,3);           //permiso para interrupción portb
 }
 set_bit(intcon,7);            //permiso para interrupciónes
}
```

```
void c_campo(void)          //cambia de campo
{
 if(campo == 1)             //si está en campo impar
 {
  campo = 0;                //pasa a campo par
  switch(modo)              //determina modo de funcionamiento
  {
   case 1:                  //si está en modo 1
        linea = 336;        //1ª linea campo par modo 1
        break;              //retorna
   case 2:                  //si está en modo 2
        linea = 321;        //1ª linea campo par modo 2
        break;              //retorna
   case 3:                  //si está en modo 3
        linea = 330;        //1ª linea campo par modo 3
        break;              //retorna
   case 4:                  //si está en modo 4
        linea = 318;        //1ª linea campo par modo 4
  }
 }
 else                       //si está en campo par
 {
  campo = 1;                //pasa a campo impar
  switch(modo)              //determina modo de funcionamiento
  {
   case 1:                  //si está en modo 1
        linea = 23;         //1ª linea campo impar modo 1
        break;              //retorna
   case 2:                  //si está en modo 2
        linea = 8;          //1ª linea campo impar modo 2
        break;              //retorna
   case 3:                  //si está en modo 3
        linea = 17;         //1ª linea campo impar modo 3
        break;              //retorna
   case 4:                  //si está en modo 4
        linea = 6;          //1ª linea campo impar modo 4
  }
 }
 act_campo();               //actualiza campo
 act_linea();               //actualiza linea
 cal_tmr1();                //calcula valor de tmr1
 buzzer();                  //buzzer con cambio de campo manual
}
```

```
void s_linea(void)              //aumenta número de linea
{
 linea = linea + suma;          //aumenta línea
 switch(modo)                   //determina modo de funcionamiento
 {
  case 1:                       //si está en modo 1
       if(campo == 1)           //si está en campo impar
       {
        if(linea > 310)         //si ultima linea del campo superada
        {
         campo = 0;             //campo par
         linea = 336;           //1ª linea campo par
         buzzer();              //buzzer con cambio de campo automático
        }
       }
       else                     //si está en campo par
       {
        if(linea > 623)         //si ultima linea del campo superada
        {
         campo = 1;             //campo impar
         linea = 23;            //1ª linea campo impar
         buzzer();              //buzzer con cambio de campo automático
        }
       }
       break;                   //retorna
  case 2:                       //si está en modo 2
       switch(linea)            //determina linea
       {
        case 17:                //si linea VIT
             linea = 19;        //salta lineas VIT
             break;             //retorna
        case 23:                //si ultima linea campo impar
             campo = 0;         //campo par
             linea = 321;       //1ª linea campo par
             buzzer();          //buzzer con cambio de campo automático
             break;             //retorna
        case 330:               //si linea VIT
             linea = 332;       //salta lineas VIT
             break;             //retorna
        case 336:               //si ultima linea campo par
             campo = 1;         //campo impar
             linea = 8;         //1ª linea campo impar
             buzzer();          //buzzer con cambio de campo automático
       }
       break;                   //retorna
```

```
case 3:                     //si está en modo 3
      switch(linea)         //determina linea
      {
       case 19:             //si ultima linea campo impar
            campo = 0;      //campo par
            linea = 330;    //1ª linea campo par
            buzzer();       //buzzer con cambio de campo automático
            break;          //retorna
       case 332:            //si ultima linea campo par
            campo = 1;      //campo impar
            linea = 17;     //1ª linea campo impar
            buzzer();       //buzzer con cambio de campo automático
      }
      break;                //retorna
 case 4:                    //si está en modo 4
      if(campo == 1)        //si está en campo impar
      {
       if(linea > 310)      //si ultima linea del campo superada
       {
        campo = 0;          //campo par
        linea = 318;        //1ª linea campo par
        buzzer();           //buzzer con cambio de campo automático
       }
      }
      else                  //si está en campo par
      {
       if(linea > 623)      //si ultima linea del campo superada
       {
        campo = 1;          //campo impar
        linea = 6;          //1ª linea campo impar
        buzzer();           //buzzer con cambio de campo automático
       }
      }
}
act_campo();                //actualiza campo
act_linea();                //actualiza linea
cal_tmr1();                 //calcula valor de tmr1
}
```

```
void b_linea(void)              //disminuye número de linea
{
 linea = linea - suma;          //disminuye línea
 switch(modo)                   //determina modo de funcionamiento
 {
  case 1:                       //si está en modo 1
       if(campo == 1)           //si está en campo impar
       {
        if(linea < 23)          //si primera linea del campo superada
        {
         campo = 0;             //campo par
         linea = 623;           //ultima linea campo par
         buzzer();              //buzzer con cambio de campo automático
        }
       }
       else                     //si está en campo par
       {
        if(linea < 336)         //si primera linea del campo superada
        {
         campo = 1;             //campo impar
         linea = 310;           //ultima linea campo impar
         buzzer();              //buzzer con cambio de campo automático
        }
       }
       break;                   //retorna
  case 2:                       //si está en modo 2
       switch(linea)            //determina linea
       {
        case 7:                 //si primera linea campo impar
               campo = 0;       //campo par
               linea = 335;     //ultima linea campo par
               buzzer();        //buzzer con cambio de campo automático
               break;           //retorna
        case 331:               //si linea VIT
               linea = 329;     //salta lineas VIT
               break;           //retorna
        case 320:               //si primera linea campo par
               campo = 1;       //campo impar
               linea = 22;      //ultima linea campo impar
               buzzer();        //buzzer con cambio de campo automático
               break;           //retorna
        case 18:                //si linea VIT
               linea = 16;      //salta lineas VIT
       }
       break;                   //retorna
```

```
case 3:                     //si está en modo 3
      switch(linea)         //determina linea
      {
       case 16:             //si primera linea campo impar
            campo = 0;      //campo par
            linea = 331;    //ultima linea campo par
            buzzer();       //buzzer con cambio de campo automático
            break;          //retorna
       case 329:            //si primera linea campo par
            campo = 1;      //campo impar
            linea = 18;     //ultima linea campo impar
            buzzer();       //buzzer con cambio de campo automático
      }
      break;                //retorna
 case 4:                    //si está en modo 4
      if(campo == 1)        //si está en campo impar
      {
       if((linea == 5) || (linea > 627))//si primera linea del campo superada
       {
        campo = 0;          //campo par
        linea = 623;        //ultima linea campo par
        buzzer();           //buzzer con cambio de campo automático
       }
      }
      else                  //si está en campo par
      {
       if(linea < 318)      //si primera linea del campo superada
       {
        campo = 1;          //campo impar
        linea = 310;        //1ª linea campo impar
        buzzer();           //buzzer con cambio de campo automático
       }
      }
 }
 act_campo();               //actualiza campo
 act_linea();               //actualiza linea
 cal_tmr1();                //calcula valor de tmr1
}


void act_campo(void)        //actualiza campo en pantalla
{
 lcd_coman(0xCA);           //comando linea 2 posición 11
 if(campo == 1)             //si está en campo impar
  cadena("IMPAR");          //actualiza mensaje en pantalla
 else                       //si está en campo par
  cadena("PAR  ");          //actualiza mensaje en pantalla
}
```

```
void act_linea(void)            //actualiza linea en pantalla
{
 char linea2;                   //define una variable para el calculo de caracteres
 centenas = linea/100;          //calcula centenas(centenas = parte entera división)
 linea2 = linea%100;            //linea2 = resto división anterior
 decenas = linea2/10;           //calcula decenas(decenas = parte entera división)
 unidades = linea2%10;          //calcula unidades(unidades = resto división anterior)
 lcd_coman(0x9A);               //comando linea 3 posición 11
 lcd_write(centenas + 48);      //escribe centenas
 lcd_write(decenas + 48);       //escribe decenas
 lcd_write(unidades + 48);      //escribe unidades
}


void act_suma(void)             //actualiza sumador en pantalla
{
 if((modo == 1) || (modo == 4))     //si está en modo 1 ó modo 4 -> actualiza
 {
  if(suma == 1)                 //si sumador = 1
  {
   suma = 10;                   //sumador = 10
   lcd_coman(0xDB);             //comando linea 4 posición 11
   cadena("+10");               //actualiza mensaje en pantalla
  }
  else                          //si sumador = 10
  {
   suma = 1;                    //sumador = 1
   lcd_coman(0xDB);             //comando linea 4 posición 11
   cadena(" +1");               //actualiza mensaje en pantalla
  }
 buzzer();                      //buzzer con cambio de sumador
 }
}


void cal_tmr1(void)             //calcula el contenido de tmr1
{
 if(campo == 1)                 //si está en campo impar
  tmr1 = 65535 - linea;         //número de impulsos para desbordamiento
 else                           //si está en campo par
  tmr1 = 65848 - linea;         //número de impulsos para desbordamiento
 tmr1b = tmr1 & 0x00FF;         //calcula lsb de tmr1
tmr1a = (tmr1 & 0xFF00)/256; //calcula msb de tmr1
}
```

```
void buzzer(void)                //buzzer
{
 char i = 0;                     //define variable para contaje
 for(i=0; i<100; i++)            //señal rectangular para alerta sonora
 {
  set_bit(portc,2);
  delay_us(250);
  clear_bit(portc,2);
  delay_us(250);
 }
}
```

Modulo LCD.

```
//Conexión de la pantalla LCD LM041L al Microcontrolador PIC16F876.
//Bits 0-2 de la puerta A bus de control.
//Bits 0-7 de la puerta B bus de datos.


//Inclusión de librerías

#include "def.h"


//Declaración de funciones y variables

void lcd_setup(void);
void lcd_coman(char com);
void lcd_write(char dat);
void cadena(const char *cad);
void menu(const char *cad1);
void inicio(void);
void menu_p(void);
void menu_1(void);
void menu_2(void);
void menu_3(void);
void menu_4(void);


//Declaración de funciones y variables externas

extern void buzzer(void);
extern char modo, campo, suma;
extern int linea;
```

```
//Definición de funciones

void lcd_setup(void)          //configura e inicializa lcd
{
 delay_ms(50);                //espera encendido de lcd
 clear_bit(porta,lcd_en);     //mantiene lcd desabilitada
 clear_bit(porta,lcd_rw);     //mantiene modo escritura
 lcd_coman(system_set);       //envia comando interfaz 8 bits, pantalla 4 lineas, fuente 5x7 puntos
 lcd_coman(display_on);       //envia comando pantalla on
}


void lcd_coman(char com)      //envia comando a lcd
{
 clear_bit(porta,lcd_rs);     //selecciona modo comandos
 trisb = 0x00;                //configura puerta b para bus lcd(bits 7-0 out)
 lcd_write(com);              //escribe comando
 delay_ms(5);                 //espera por si lcd ocupada
 set_bit(porta,lcd_rs);       //mantiene modo datos
}


void lcd_write(char dat)      //escribe comando o dato en la puerta b
{
 portb = dat;                 //escribe en la puerta b
 delay_ms(1);                 //espera por si lcd ocupada
 set_bit(porta,lcd_en);       //habilita lcd
 clear_bit(porta,lcd_en);     //deshabilita lcd
}


void cadena(const char *cad) //escribe cadena en pantalla
{
 char i = 0;                  //define una variable para el contaje
 while(cad[i] != 0)           //ejecuta bucle mientras no termine la cadena
 {
  lcd_write(cad[i++]);        //envia caracter de la cadena
 }
}


void menu(const char *cad1) //escribe menu en pantalla
{
 lcd_coman(clear_lcd);        //envia comando borra pantalla y cursor a casa
 char i = 0;                  //define una variable para el contaje
```

```
while(cad1[i] != 0)          //ejecuta bucle mientras no termine la cadena
 {
 lcd_write(cad1[i++]);       //envia caracter de la cadena
 switch(i)                   //determina posición del cursor
 {
  case 16:                   //si fin linea 1
       lcd_coman(line2);     //linea 2
       break;
  case 32:                   //si fin linea 2
       lcd_coman(line3);     //linea 3
       break;
  case 48:                   //si fin linea 3
       lcd_coman(line4);     //linea 4
       break;
 }
 }
}


void inicio(void)            //mensajes de inicio
{
 menu("  UNIVERSIDAD     DE HUELVA     I.T.INDUSTRIAL   ELECTRONICA   ");
                             //actualiza menu
 delay_s(4);                 //espera para lectura
 menu("   GENERADOR-      -MONITOR        DE VIDEO        PARA TV     ");
                             //actualiza menu
 delay_s(4);                 //espera para lectura
 menu(" HARDWARE: v.1.0              SOFTWARE: v.1.0              ");
                             //actualiza menu
 delay_s(4);                 //espera para lectura
 menu("AUTOR: JOSE JUAN BAUTISTA PULIDODIRECTOR: ANDRÉS ROLDÁN ARANDA");
                             //actualiza menu
 delay_s(4);                 //espera para lectura
}


void menu_p(void)            //menu principal
{
 menu("-> MODO VIDEO   -> MODO T.X.T.  -> MODO V.I.T.  -> MODO CONTINUO");
                             //actualiza menu
 modo = 0;                   //actualiza modo
 buzzer();                   //buzzer al entrar en menu principal
}
```

```
void menu_1(void)            //menu 1
{
 menu("<- MODO VIDEO  <> CAMPO: IMPAR +  LINEA: 023  -        +1<>");
                             //actualiza menu
 modo = 1;                   //actualiza modo
 campo = 1;                  //actualiza campo
 linea = 23;                 //actualiza linea
 suma = 1;                   //actualiza sumador
 cal_tmr1();                 //calcula valor de tmr1
 buzzer();                   //buzzer al entrar en modo 1
}


void menu_2(void)            //menu 2
{
 menu("<- MODO T.X.T.  <> CAMPO: IMPAR +  LINEA: 008  -           ");
                             //actualiza menu
 modo = 2;                   //actualiza modo
 campo = 1;                  //actualiza campo
 linea = 8;                  //actualiza linea
 suma = 1;                   //actualiza sumador
 cal_tmr1();                 //calcula valor de tmr1
 buzzer();                   //buzzer al entrar en modo 2
}


void menu_3(void)           //menu 3
{
 menu("<- MODO V.I.T.  <> CAMPO: IMPAR +  LINEA: 017  -           ");
                             //actualiza menu
 modo = 3;                   //actualiza modo
 campo = 1;                  //actualiza campo
 linea = 17;                 //actualiza linea
 suma = 1;                   //actualiza sumador
 cal_tmr1();                 //calcula valor de tmr1
 buzzer();                   //buzzer al entrar en modo 3
}
```

```
void menu_4(void)           //menu 4
{
 menu("<- MODO CONTINUO<> CAMPO: IMPAR +  LINEA: 006   -         +1<>");
                            //actualiza menu
 modo = 4;                  //actualiza modo
 campo = 1;                 //actualiza campo
 linea = 6;                 //actualiza linea
 suma = 1;                  //actualiza sumador
 cal_tmr1();                //calcula valor de tmr1
 buzzer();                  //buzzer al entrar en modo 4
}
```

Definiciones.

//Definiciones necesarias para el Microcontrolador y la LCD

//Definición de registros del Microcontrolador

```
char tmr0@0x01;             //registro tmr0
char status@0x03;           //registro estado
char porta@0x05;            /registro puerta a
char portb@0x06;            //registro puerta b
char portc@0x07;            //registro puerta c
char intcon@0x0B;           //registro interrupciones
char tmr1l@0x0E;            //registro tmr1l
char tmr1h@0x0F;            //registro tmr1h
char pir1@0x0C;             //registro banderas interrupción periféricos
char t1con@0x10;            //registro configuración tmr1
char tmr2@0x11;             //registro tmr2
char t2con@0x12;            //registro configuración tmr2
char option@0x81;           //registro opciones
char trisa@0x85;            //registro triestado puerta a
char trisb@0x86;            //registro triestado puerta b
char trisc@0x87;            //registro triestado puerta c
char pie1@0x8C;             //registro habilitación interrupciones periféricos
char adcon1@0x9F;           //registro configuración periféricos puerta a
```

//Definiciones para el control de la LCD

```
#define lcd_rs    0x00      //bit 0 puerta a(0=comandos, 1=caracteres)
#define lcd_rw    0x01      //bit 1 puerta a(0=escritura, 1=lectura)
#define lcd_en    0x02      //bit 2 puerta a(0=lcd desactivado, 1=lcd activado)
```

```
//Definición de comandos de la LCD

#define clear_lcd           0x01   //borra pantalla
#define cursor_home         0x02   //cursor a casa
#define entry_mode          0x06   //modo introducción, inc. pos. cursor
#define entry_mode_shift    0x07   //-con desplazamiento
#define display_on          0x0C   //pantalla on
#define display_off         0x08   //pantalla off
#define cursor_on           0x0F   //cursor on parpadea
#define cursor_off          0x0C    //cursor off
#define system_set          0x38   //interfaz 8 bits, pantalla 4 lineas, fuente 5x7 puntos
#define line1               0x80   //linea 1 posición 1
#define line2               0xC0   //linea 2 posición 1
#define line3               0x90   //linea 3 posición 1
#define line4               0xD0   //linea 4 posición 1
```

CAPITULO 6:
DISEÑO, FABRICACIÓN Y MONTAJE DE PCB'S.

DISEÑO, FABRICACIÓN Y MONTAJE DE PCB'S

Para la obtención de las diferentes placas de circuito impreso utilizadas en el proyecto (todas ellas de primera calidad) se han de seguir una serie de pasos imprescindibles. Estos pasos son los siguientes:

- Diseño mediante computadora – CAD.
- Impresión del fotolito.
- Fabricación.
  - Insoladora.
  - Revelado.
  - Ataque químico.
  - Limpiado.
- Montaje.
  - Cortado.
  - Taladrado.
  - Soldadura.
  - Lacado.

6.1. DISEÑO.

Hasta hace poco tiempo, el proceso de fabricación de las placas de circuito impreso era lento, difícil y laborioso. Al introducirse el ordenador en el mundo del

diseño electrónico han surgido programas, entre ellos Protel 98, que facilitan y aceleran el proceso de fabricación.

Protel98 consta de cuatro funciones para las siguientes tareas específicas:

1) SCH. - Captura de esquemas de circuitos electrónicos.
2) SCHLib. - Captura de componentes para esquemas.
3) PCB. - Generación de placas de circuito impreso.
4) PCBLib. – Generación de componentes para PCB'S.

6.1.1. Protel 98 - PCB.

Protel 98 – PCB es un programa de altas prestaciones para diseñar y producir placas de circuito impreso (Printed Circuit Board o PCB).

Dispone de herramientas como:

- Rules: Permite establecer reglas en el diseño, como por ejemplo, anchura máxima y mínima de las pistas, separación mínima entre pistas, tamaño de pads y vias, caras utilizadas, etc.
- Design rules chek: Permite comprobar las reglas de diseño previamente establecidas.
- Auto – Place: Coloca automáticamente los componentes según unas reglas previamente establecidas.
- Auto – Route: Teniendo en cuenta las reglas de diseño y las estrategias escogidas, traza automáticamente las pistas del circuito.

En nuestro caso, tanto la ubicación de los componentes como el trazado de las pistas se han realizado de forma manual. El programa está destinado al uso profesional, y toda la potencia del mismo solo puede obtenerse con procedimientos y herramientas del mismo nivel, por tanto, para la realización de las placas con tecnología casera es necesario imponer serias limitaciones, siendo las más importantes el tamaño de las pistas, vias y pads, y el número de caras.

El programa dispone de múltiples formatos de salida, permitiendo sacar el resultado por una impresora o plotter, convertir el resultado en un fichero de formato industrial Gerber para fotoimpresión y guardar el diseño en un disquete o en el disco duro.

Por último, se muestran algunos ejemplos de los resultados obtenidos con las diferentes herramientas.

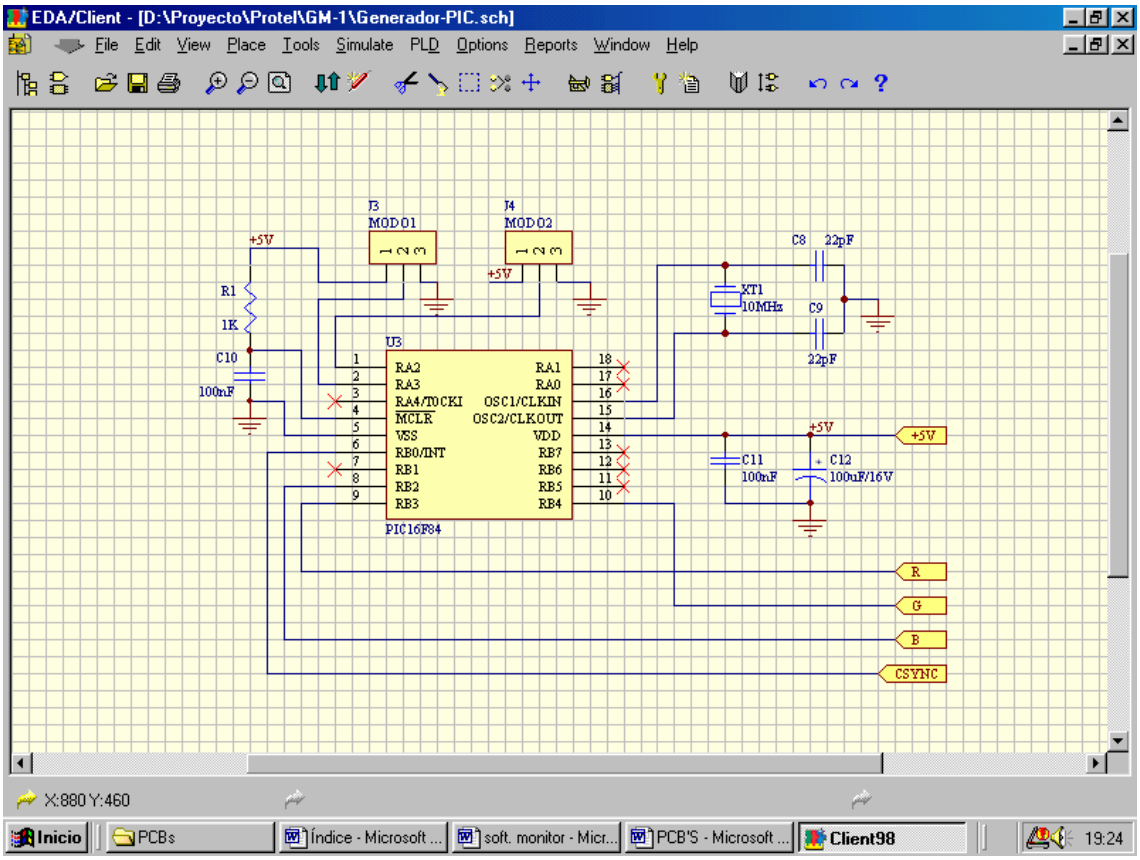


Figura 6.1 – Esquema de conexión del microcontrolador del generador de vídeo.

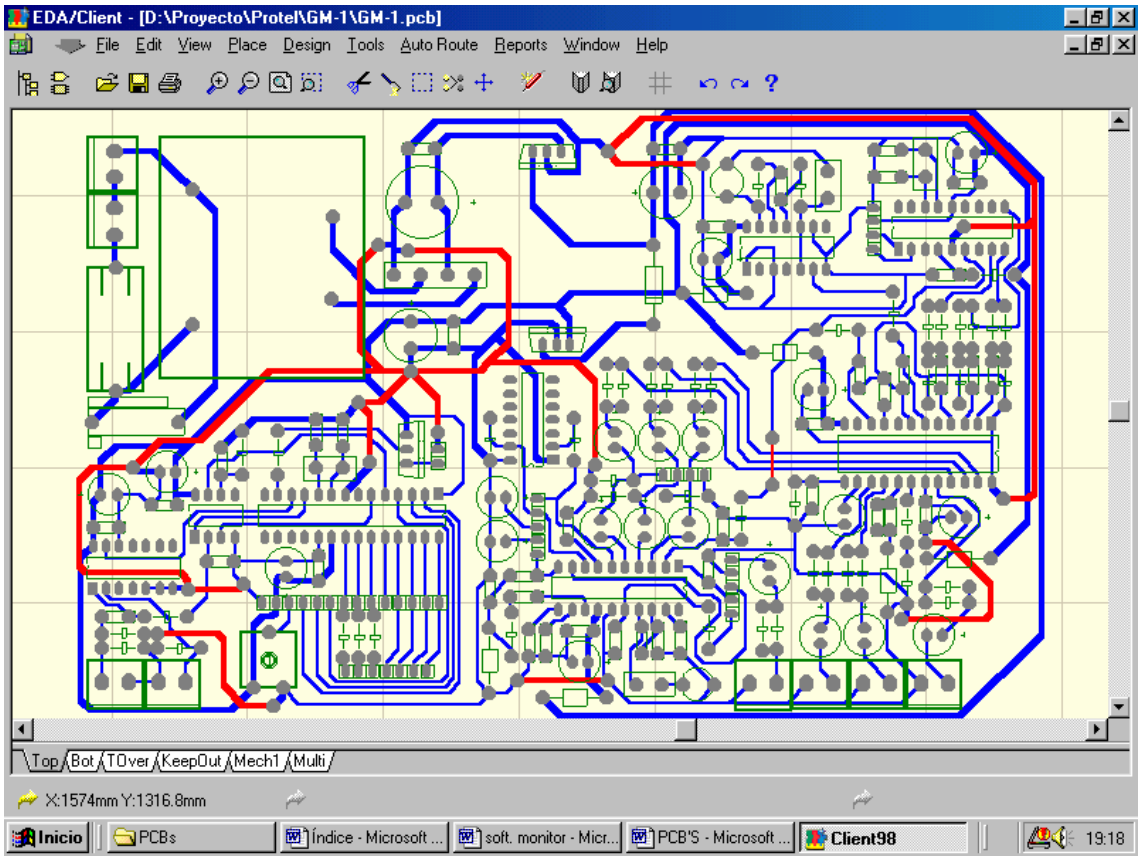


Figura 6.2 – Pistas de la placa definitiva.

6.2. FOTOLITO.

La calidad del fotolito que utilicemos en la insolación, incidirá directamente en la calidad del circuito que obtendremos. El soporte sobre el que debe realizarse el circuito debe de ser totalmente transparente o permeable a la luz U.V., mientras que el dibujo del circuito tendrá que estar realizado con tinta opaca a los rayos U.V. Para la realización del fotolito, se imprimió el circuito de la figura 6.2 (incluyendo el relleno necesario) en papel transparente mediante una impresora tipo "láser". El resultado final puede verse en el capítulo de planos.

6.3. FABRICACIÓN DE LA PLACA.

La fabricación de los circuitos impresos se ha llevado a cabo de forma manual. Las placas de los prototipos se han realizado a una cara, mientras que la placa definitiva va a dos caras. Todas ellas se obtienen a partir de un positivo de la cara de las pistas (fotolito). Se usa una placa de circuito con película presensibilizada positiva (sensible a la luz), constituida por una base de resina epoxi y fibra de vidrio cubierta con una lámina de cobre por cada una de sus caras (1 o 2 según el caso). Las características de las placas, se expresan en la tabla 6.1.

Espesor de la resina fotosensible	2,5 µm.
Espesor del estratificado FR4	16/10ª
Espesor de la baquelita	15/10ª
Espesor de la lámina de cobre	35 µm.
Cumple las normas	"MIL 13949 F" y "UL 94 Vo"

Tabla 6.1 - Características técnicas de las placas fotosensibilizadas.

El procedimiento que se ha utilizado para la realización de las placas, es el siguiente:

6.3.1. INSOLADO.

Una vez retirada la protección adhesiva de la placa, se posiciona el fotolito o dibujo original sobre la cara fotosensible de la placa. A continuación se coloca la unión placa + fotolito en la insoladora durante un tiempo. Este tiempo de exposición vendrá determinado por las características del epoxi fotosensible, por lo que se han de seguir las referencias que el fabricante incluye en la placa. En este caso el tiempo de exposición fue de 2,30 minutos.

6.3.2. REVELADO.

Después del insolado y sin exponer la placa a la luz, se ha introducido en una disolución de revelador sólido en agua según las proporciones indicadas por el fabricante; hasta que a simple vista se apreciaron las pistas. Mientras se ejecuta este proceso, el revelador se va oscureciendo.

Toda la resina fotosensible que haya sido insolada desaparecerá en menos de dos minutos. Una vez que se vean bien las pistas, se lavará la placa con agua limpia común.

6.3.3. ATAQUE QUÍMICO.

Después de revisada la placa. Se procede al ataque químico para eliminar el cobre de las zonas no protegidas de la resina. En un tiempo aproximado de 2 minutos, deberá quedar grabado el circuito. Hay que cuidar el tiempo de atacado, pues un exceso del mismo haría desaparecer el circuito o parte de él. La disolución empleada está compuesta por: $H_2O_2(200ml.)+Agua\ fuerte(200ml)+H_2O(100ml)$.

6.3.4. LIMPIADO.

Una vez realizado el ataque químico se procederá a limpiar la placa, utilizando primero agua limpia común para eliminar el ácido que pudiera haber quedado de la etapa anterior y acto seguido se frota con un algodón impregnado en alcohol para eliminar los restos de resina.

6.4. - MONTAJE DE LA PLACA.

Para el montaje de los componentes en la placa se han seguido los siguientes pasos:

6.4.1. CORTADO.

Con una cizalla plana se corta la porción de placa que ha quedado impresa con las pistas, dejando unos márgenes suficientes.

6.4.2. TALADRADO.

Los taladros se realizan desde la cara del cobre. El centro del taladro corresponde al centro de las islas de soldadura. Los taladros para los terminales tipo espadín son de 1.25 a 1.3 mm. de diámetro, 0.8 mm. para los circuitos integrados, de

1mm. para los componentes de uso corriente tales como resistencias, condensadores ..., y de 3.5mm. para los tornillos de sujeción.

6.4.3. SOLDADURAS.

Se realizarán con estaño enriquecido con resinas sobre las islas de cobre que finalizan o interceptan cada pista. Para ello hemos utilizado los siguientes elementos:

- Estaño: Este metal se presenta para su comercialización, en forma de carrete de hilo metálico dúctil, en cuya proporción interviene el estaño y el plomo en determinadas proporciones (60% de estaño y 40% de plomo). Se ha utilizado un hilo de estaño de sección reducida provisto de varios conductos internos llenos de un tipo especial de resina que garantiza la eficacia de las soldaduras.
- Soldador: Mediante esta herramienta se proporciona el calor necesario para fundir el estaño y hacer que éste corra líquido por la superficie a soldar. Debido a la naturaleza de la aplicación se utilizó un soldador de tipo "lápiz", de 15 W. de potencia, alimentado con una tensión alterna de 220 V. La punta de soldadura es de cobre, a la que se ha aplicado un tratamiento antioxidante.

6.4.4. LACADO ANTICORROSIÓN.

Para evitar que la placa se oxide con el tiempo, se aplica a la cara de pistas de cobre una capa de laca aislante y protectora para uso en PCBs. Las características de la laca utilizada son las siguientes:

- Protege los componentes y evita la oxidación de las pistas de cobre.
- No es necesario eliminarla antes de soldar.
- Rigidez dieléctrica: 40 kv / mm.
- Constante dieléctrica a 10 KHz: 2,83.
- Pérdidas dieléctricas: 0,01 a 1 MHz.
- Resistencia volumétrica: 10^{15} Ωcm.
- Temperatura de trabajo de la capa aplicada: de –20ºC a +150ºC.

CAPITULO 7:
PLANOS.

CAPITULO 8:
PRESUPUESTOS.

PRESUPUESTOS

PRECIO DESCOMPUESTO			
GENERADOR – MONITOR DE VÍDEO			
CANTIDAD	CONCEPTO	PRECIO UNIDAD	IMPORTE
1	Condensador Electrolítico 3300uF/ 25V	0,49	0,49
2	Condensador Electrolítico 1000uF/ 25V	0,33	0,66
9	Condensador Electrolítico 220uF / 16V	0,12	1,08
3	Condensador Electrolítico 100uF / 16V	0,09	0,27
3	Condensador Electrolítico 47uF / 16V	0,09	0,27
3	Condensador Electrolítico 22uF / 16V	0,09	0,27
3	Condensador Electrolítico 10uF / 16V	0,21	0,63
1	Condensador Poliéster 100nF / 400V	0,30	0,30
20	Condensador Poliéster 100nF / 63V	0,12	2,40
5	Condensador Poliéster 10nF / 63V	0,12	0,60
3	Condensador Poliéster 1nF / 63V	0,12	0,36
1	Condensador Disco 560pF	0,09	0,09
2	Condensador Disco 220pF	0,09	0,18
1	Condensador Disco 56pF	0,09	0,09

PRECIO DESCOMPUESTO			
GENERADOR – MONITOR DE VÍDEO			
CANTIDAD	CONCEPTO	PRECIO UNIDAD	IMPORTE
2	Condensador Disco 47pF	0,09	0,18
1	Condensador Disco 39pF	0,09	0,09
4	Condensador Disco 22pF	0,09	0,36
2	Condensador Disco 4.7pF	0,09	0,18
1	Condensador Disco 2.2pF	0,09	0,09
2	Condensador Variable 5 – 25pF	0,42	0,84
36	Resistencia 1/4W	0,03	1,08
1	Resistencia Variable 1/2W	0,18	0,18
2	Potenciómetro 1W	0,90	1,80
1	Diodo 1N4007	0,06	0,06
1	Puente Rectificador B380C1500	0,60	0,60
1	Regulador 7812 TO220	0,36	0,36
1	Regulador 7805 TO220	0,36	0,36
2	Disipadores TO220	0,34	0,68
1	Circuito Integrado PIC16F84-10	9,92	9,92
1	Circuito Integrado 74HC04	0,30	0,30
1	Circuito Integrado CXA1645P	8.34	8.34
1	Circuito Integrado MC1377P	3.34	3.34
1	Circuito Integrado PIC16F876-04	8,28	8,28
1	Circuito Integrado TC4066BP	0,27	0,27
1	Circuito Integrado LM1881N	4,45	4,45

PRECIO DESCOMPUESTO			
GENERADOR – MONITOR DE VÍDEO			
CANTIDAD	CONCEPTO	PRECIO UNIDAD	IMPORTE
1	Zócalo 28 Pines	0,26	0,26
1	Zócalo 18 Pines	0,13	0,13
1	Zócalo 16 Pines	0,13	0,13
2	Zócalo 14 Pines	0,13	0,26
1	Zócalo 8 Pines	0,13	0,13
4	Tira Pines Torneados	0,64	2,56
1	Cristal Cuarzo 4MHz	1,29	1,29
2	Cristal Cuarzo 4.43MHz	1,29	2,58
1	Cristal Cuarzo 10MHz	1,29	1,29
10	Bobina Miniatura 22uH	0,48	4,80
1	Cable Alimentación Tipo 8 Homologado	0,77	0,77
1	Interruptor de Red	0,65	0,65
1	Portafusible	0,77	0,77
5	Fusible 50mA	0,18	0,90
1	Transformador 12V / 5VA	3,36	3,36
8	Regleta conexión 2 polos	0,21	1,68
1	Transistor BC337	0,09	0,09
1	Altavoz 8Ω / 30mW	1,72	1,72
6	Pulsador Miniatura Abierto	0,36	2,16
1	Display LM041L	28	28
6	Conector BNC Hembra Base	0,78	4,68
6	Interruptores 2 Posiciones	0,65	3,9
2	Mando para Potenciómetro	0,30	0,60
1	Caja RETEX Solbox 13	16,19	16,19
4	Separadores Metálicos	0,13	0,52
16	Tornillos 2mm	0,03	0,48

PRECIO DESCOMPUESTO	
GENERADOR – MONITOR DE VÍDEO	
IMPORTE	126,35 €
16% I.V.A.	20,21 €
TOTAL FACTURA (EUROS)	146,56 €
TOTAL FACTURA (PESETAS)	24.388 Pts

PRECIO DESCOMPUESTO			
MATERIALES Y COMPLEMENTOS			
CANTIDAD	CONCEPTO	PRECIO UNIDAD	IMPORTE
1	KP-27 Placa Fibra 2 Caras 200 X 300 Positiva	12,84	12,84
1	Sobre Revelador Positivo	2,70	2,70
1	Bote Agua Fuerte	1,02	1,02
1	Bote Agua Oxigenada 110Vol	5,70	5,70
1	Bote Alcohol	0,90	0,90
4	Broca Metal 0.8 mm	0,90	3,60
1	Broca Metal 1 mm	0,90	0,90
1	Rollo Estaño	1,44	1,44
5	Cable 2 x 1.5mm Rojo	0,18	0,90
5	Cable 2 x 1.5mm Negro	0,18	0,90
IMPORTE			30,9 €
16 % I.V.A.			4,94 €
TOTAL FACTURA (EUROS)			35,84 €
TOTAL FACTURA (PESETAS)			5.964 Pts

PRECIO DESCOMPUESTO			
MANO DE OBRA			
HORAS	CONCEPTO	PRECIO UNIDAD	IMPORTE
50	Diseño del Hardware	15	750
150	Programación del Software	15	2.250
50	Fabricación y Montaje	10	500
12	Verificación y Comprobación	20	240
50	Horario Facultativo por Redacción del Proyecto	10	500
IMPORTE			**4.240 €**
16 % I.V.A.			**678,4 €**
TOTAL FACTURA (EUROS)			**4.918,4 €**
TOTAL FACTURA (PESETAS)			**818.421,76 Pts**

PRESUPUESTO DE EJECUCIÓN MATERIAL	
Generador – Monitor de Vídeo	126,35
Materiales y Complementos	30,9
IMPORTE	**157,25 €**
16 % I.V.A.	**25,16 €**
TOTAL FACTURA (EUROS)	**182,41 €**
TOTAL FACTURA (PESETAS)	**30.353 Pts**

PRESUPUESTO TOTAL	
Presupuesto de Ejecución Material	157,25
Mano de Obra	4.240
Control de Calidad	60
IMPORTE	**4.457,25 €**
16 % I.V.A.	**713,16 €**
TOTAL FACTURA (EUROS)	**5.170 €**
TOTAL FACTURA (PESETAS)	**860.356 Pts**

Asciende el presente proyecto a las figuradas:

CINCO MIL CIENTO SETENTA EUROS .

OCHOCIENTAS SESENTA MIL TRESCIENTAS CINCUENTA Y SEIS PESETAS.

Para hacerlo constar firma el presente proyecto el Ingeniero Técnico Industrial:

JOSÉ JUAN BAUTISTA PULIDO.

Firma

Octubre de 2002.

CAPITULO 9:
FUTURAS MEJORAS.

FUTURAS MEJORAS

Una de las ventajas más importantes que se tiene a la hora de trabajar con un microcontrolador es que se logra una gran flexibilidad en cuanto a futuras modificaciones del funcionamiento del sistema. Esto permite que con unos mínimos cambios en el *hardware* del dispositivo, el sistema desarrollará nuevas funciones con tan sólo modificar el *software*.

GENERADOR DE VÍDEO.

En el caso del generador de vídeo se podrían realizar las siguientes modificaciones:

1) Aumentar el número de patrones: Se podrían rediseñar las funciones de vídeo para hacer estas más compactas, liberando de esta forma espacio en la memoria de programa, para así poder incluir más patrones de vídeo.

2) Mejorar el teclado: El terminal 3 del PORTA se encuentra disponible en el conector del teclado. Este se podría utilizar para introducir un nuevo pulsador, de manera que un pulsador avanzaría hasta el siguiente patrón y el otro retrocedería, evitando de esta forma tener que pasar todos los patrones para volver al anterior.

3) Eliminar un oscilador de 4.43MHz: El conversor de vídeo MC1377 integra un oscilador para la subportadora de color, siendo necesarios varios componentes externos (cristal de cuarzo, condensadores). Por otro lado, el conversor de vídeo CXA1645 no incorpora oscilador, siendo necesario un oscilador externo. Entonces, la mejora consistiría en tomar la señal de 4.43MHz del conversor MC1377, y pasándola por un buffer adecuado, utilizarla para el conversor CXA 1645, ahorrándonos de esta forma el oscilador externo que se utiliza en el diseño.

4) Mejorar el control de señales: En vez de una conexión/desconexión de señales a través de interruptores mecánicos, se podría utilizar un circuito integrado de conmutación controlado por pulsadores miniatura. También se podrían controlar las señales con uno de los microcontroladores disponibles, a través de un menú adecuado en el display.

5) Introducir un control de saturación: El nivel de las componentes de color es constante, ya que el nivel de salida del microcontrolador es también constante. Se podría introducir un circuito integrado del tipo TLS1233N o LM1203N para controlar los niveles y de esta forma posibilitar nuevas configuraciones de patrones e incluso realizar procesos de ajuste.

6) Mejorar el filtro de color: La red de filtrado paso banda para la línea de información de color en el conversor MC1377, se podría mejorar con la utilización de un filtro basado en transformador TOKO. Por ejemplo con el transformador 166NNF-10264AG se consigue un ancho de banda de +/- 0.5MHz y unas perdidas de inserción de 3dB.

MONITOR DE VÍDEO.

En el caso del monitor de vídeo se podrían realizar las siguientes modificaciones:

1) Mejorar el salto de línea: En los modos de funcionamiento primero y cuarto, debido al gran número de líneas existentes, se tiene la posibilidad de cambiar las líneas de una en una y de diez en diez. Una mejora sería introducir la función de aceleración en las teclas que aumentan y disminuyen el número de línea. De esta forma, con una pulsación breve se cambiaría una línea, mientras que manteniendo pulsada la tecla se incrementaría el salto de línea automáticamente en función del tiempo que llevase la tecla pulsada.

2) Introducir nuevos modos de funcionamiento: El separador de sincronismos activo LM1881N dispone de una salida para indicar el momento en que aparece la salva de color. Esta señal se encuentra disponible en un terminal no utilizado del microcontrolador, por lo que se podría utilizar para un nuevo modo de funcionamiento, consistente en visualizar el pórtico posterior junto con la salva de color.

3) Introducir más funciones en los menús: El teclado matricial está compuesto por 5 teclas, pero permite la conexión de hasta 12 teclas. Introduciendo tres teclas más se tendrían dos columnas de cuatro teclas, una a cada lado del display, y de esta forma se podrían aumentar las funciones en los menús. Una posible función sería la del cambio automático de línea.

Por último, se podría realizar un nuevo estudio del ruido, y mejorar los filtros de alimentación y la placa de circuito impreso. El filtrado en la alimentación se puede mejorar con choques inductivos más grandes, siendo el diseño de la placa en estos casos lo más complicado.

CAPITULO 10: BIBLIOGRAFÍA Y DIRECCIONES DE INTERÉS.

BIBLIOGRAFÍA Y DIRECCIONES DE INTERÉS

MICROCONTROLADORES PIC.
E. Martín Cuenca.
J. Mª. Angulo Usategui.
I. Angulo Martínez.
EDITORIAL PARANINFO, 2000, 3ª EDICIÓN.

C++.
L. Joyanes Aguilar.
H. Castán Rodríguez.
EDITORIAL MCGRAW HILL, 1999, 1ª EDICIÓN.

MICROELECTRÓNICA.
Jacob Millman.
Arvin Gravel.
EDITORIAL HISPANO EUROPEA, 1993, 6ª EDICIÓN.

CIRCUITOS DIGITALES TTL.
R. M. Marston.
EDITORIAL PARANINFO, 1996, 1ª EDICIÓN

SISTEMAS DE TELEVISIÓN ANALÓGICOS.
Jesús García Jiménez.
E.T.S. DE INGENIEROS DE TELECOMUNICACIÓN DE MADRID, 1996, 1ª EDICIÓN

RECEPTOR DE TELEVISIÓN: PRACTICAS Y CONTROLES DE CALIDAD.
José Luis Fernández Baillo Rodríguez de Liébana.
E.T.S. DE INGENIEROS DE TELECOMUNICACIÓN DE MADRID, 1998, 4ª EDICIÓN

APUNTES DE ELECTRÓNICA DIGITAL.
Universidad de Huelva, E.P.S. "La Rábida".

APUNTES DE ELECTRÓNICA INDUSTRIAL.
Universidad de Huelva, E.P.S. "La Rábida".

APUNTES DE ELECTRÓNICA BÁSICA.
Universidad de Huelva, E.P.S. "La Rábida".

AMATEUR TELEVISION HANDBOOK.
John L. Wood, G3YQC.
Trevor Brown, G8CJS.
British Amateur Television Club, 1981-2000.

AN INTRODUCTION TO AMATEUR TELEVISION.
Mike Wooding, G6IQM.
Trevor Brown, G8CJS.
British Amateur Television Club, 1998.

http://www.comunidadelectronicos.com/

http://www.qsl.net/ik1hgi/atv.htm

http://www.fairchildsemi.com

http://www.questlink.com

CAPITULO 11:
ANEXOS.

ANEXOS

Figura 1 – Vista exterior del Generador–Monitor de Vídeo para TV.

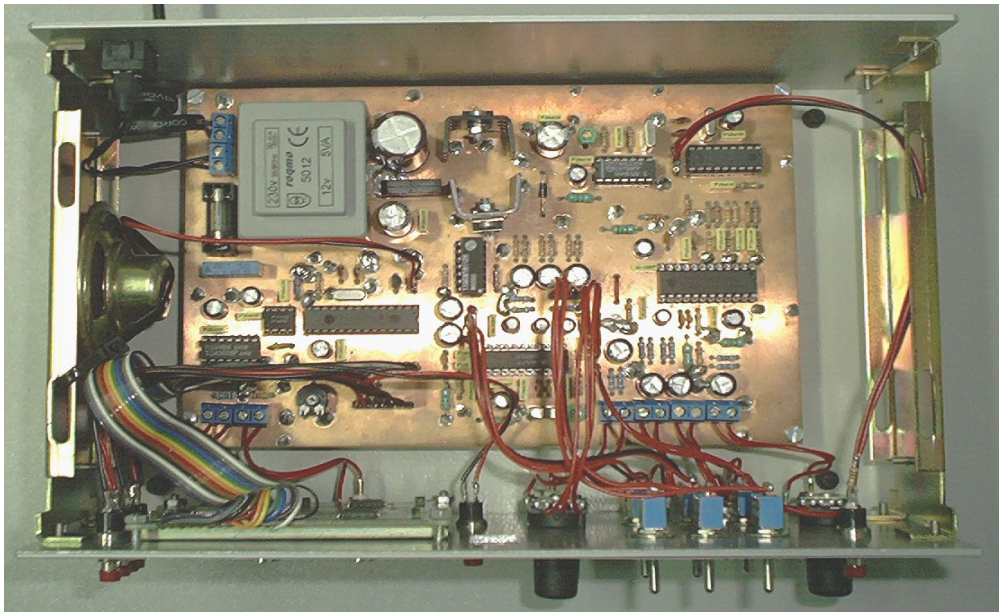


Figura 2 – Vista interior del Generador–Monitor de Vídeo para TV.

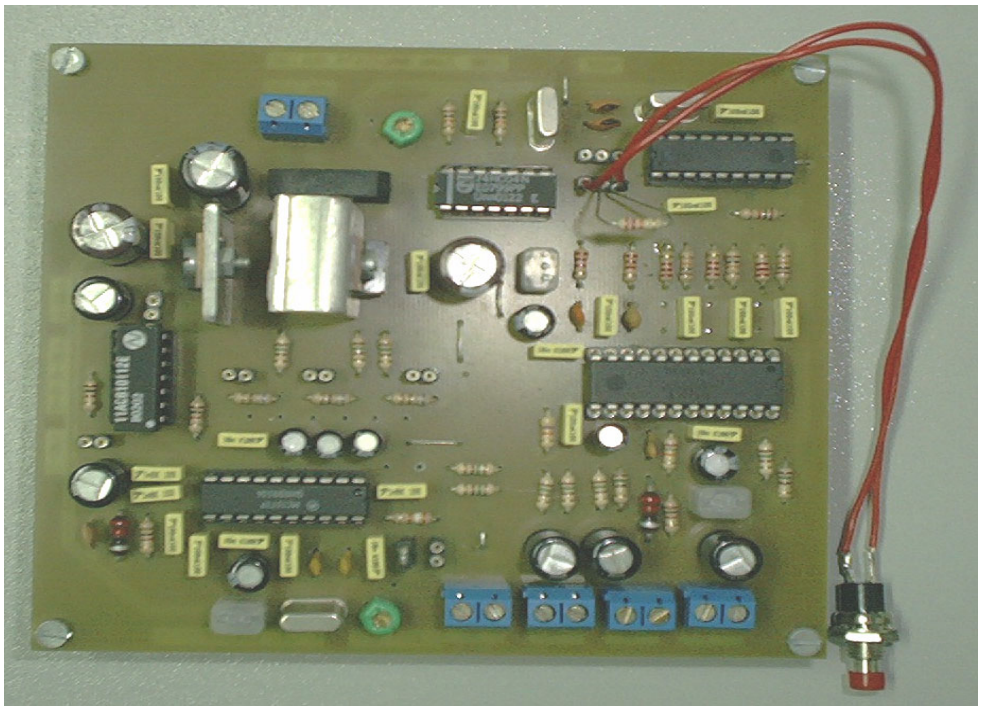


Figura 3 – Prototipo del Generador de Vídeo.

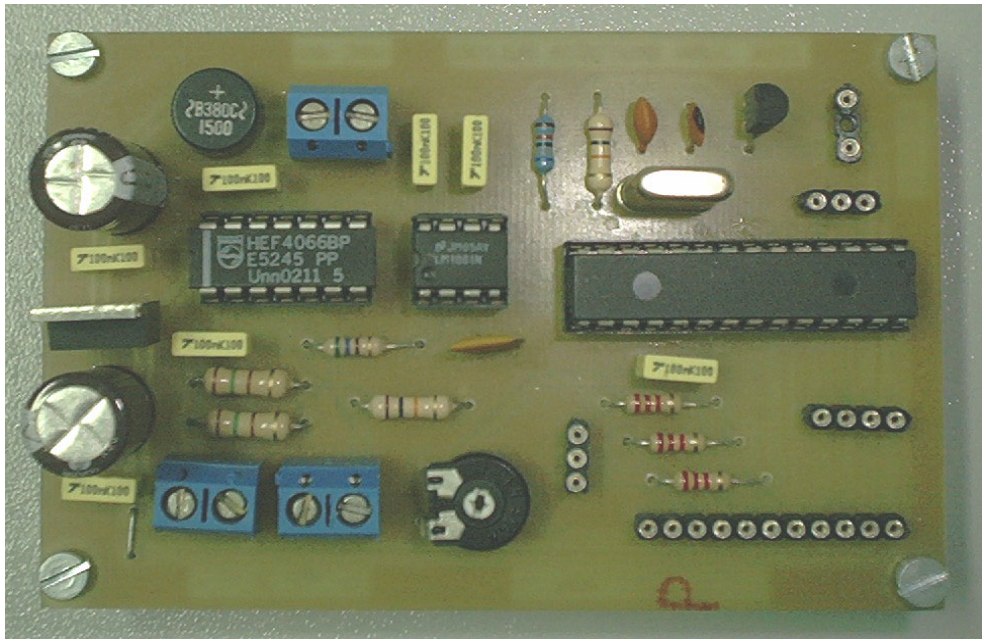


Figura 4 – Prototipo del Monitor de Vídeo.

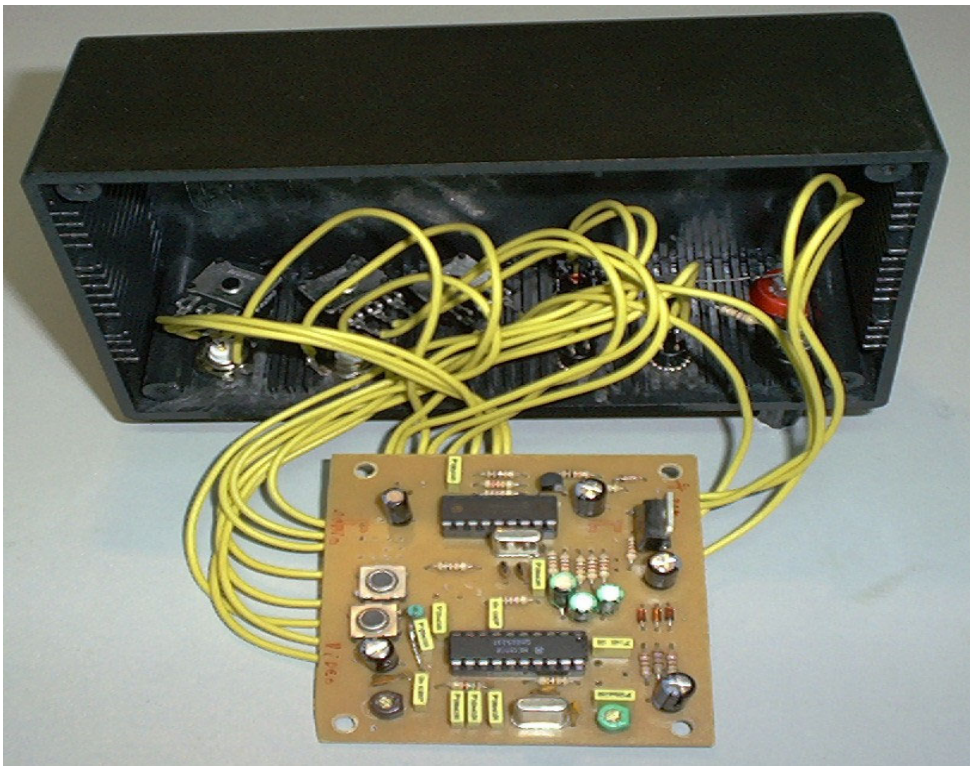


Figura 5 – Generador de Vídeo-aficionado Nº1.

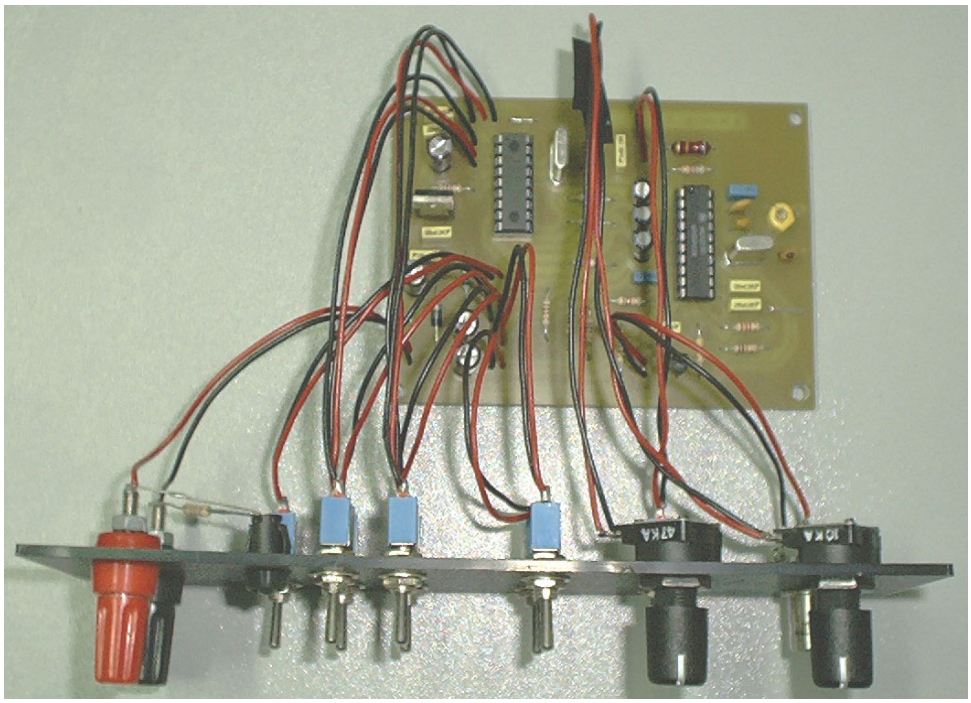


Figura 6 – Generador de Vídeo-aficionado Nº2.