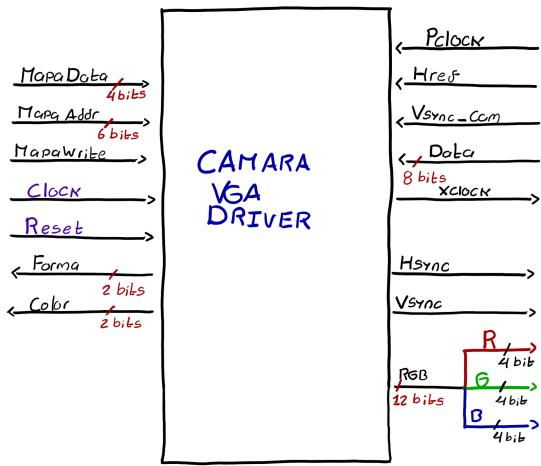


Proyecto Digital II → Camara/VGA/Driver

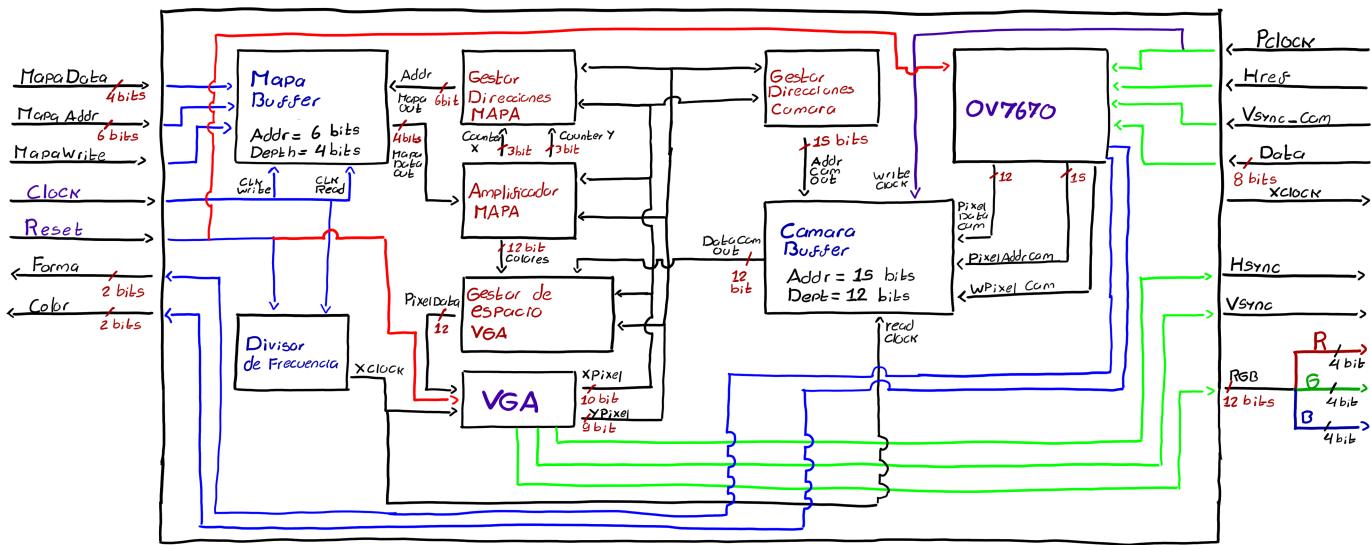
Modulo Basico (Sin Bus)



* Este modulo Combina los modulos de la Camara/Procesamiento de imagen Junto a unas memorias Y interconexiones Para organizar y mostrar todo en Pantalla, incluso Pintando un mapa a traves de una de las memoria

* Ademas de las interconexiones existen Circuitos Combinacionales Adicionales

→ Diagrama Estructural



Proyecto Digital II → Camara/VGA/Driver

Memoria (Buffer)

La memoria que se requiere implementar es del tipo lectura/escritura de doble reloj. Pues el escritor asigana una dirección, datos y la habilitación de la escritura de manera independiente Al lector que Asigna su propia dirección y datos.

→ Diagrama Funcional



Escritura

Lectura

Reg → RAM ($\text{AddrLength} \times \text{bitDepth}$), su declaración es → reg [$\text{bitDepth}-1:0$] ram [$2*\text{AddrWidth}-1:0$];
 Output Reg → DataRead (bitDepth)
 $\text{AddrLength} = 2^{\text{AddrWidth}}$

input → WriteEnable
 input → Clock R
 input → Clock W
 input → Read Addr (AddrWidth)
 input → Write Addr (AddrWidth)
 input → data_write (bitDepth)

* Para el buffer de la cámara se requiere un banco de registros de 12 bits de profundidad (Almacena formato RGB 444) y un tamaño suficiente para soportar los 176x144 pixeles del tamaño de la pantalla o sea más de 25 344 direcciones, en este caso $2^{12} = 4096$ direcciones.

* En el caso del mapa se requiere un banco de registros de 4 bits de profundidad y un tamaño suficiente para soportar el tablero 8x8, requiriendo 64 direcciones, en este caso $2^6 = 64$ direcciones.

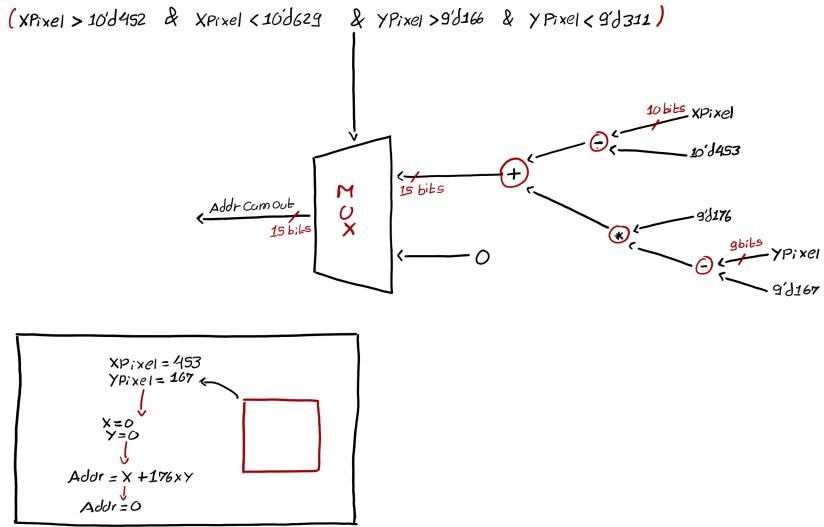
Divisor de Frecuencia

El divisor de frecuencia que se implementó es igual que los anteriores con el objetivo de lograr una frecuencia de 25 MHz, por lo cual en este caso en particular de un reloj de 50 MHz base, no se implementó un contador y se negó la señal en cada ciclo, consiguiendo un reloj de 25 MHz con un ciclo útil del 50%.

Proyecto Digital II → Camara/VGA/Driver

Gestor de direcciones de la Camara

debido a que se quiere mostrar la imagen en una posicion muy especifica de los 640×480 pixeles, mas especificamente para la region encerrada por, $452 < XPixel < 629$ y $166 < YPixel < 311$ Por lo cual se utilizará un multiplexor y sumandores, multiplicadores.



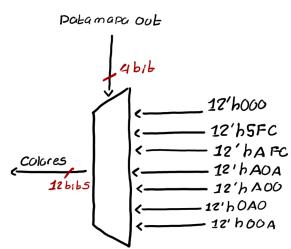
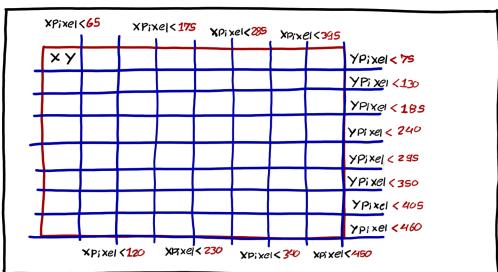
* Las sumas, restas y multiplicaciones son necesarias porque queremos transformar el XPixel 453 en 0, y YPixel 167 en 0, siendo el origen la posición inicial de la memoria

* Luego de los restos que Acomodo el punto de origen se obtiene la dirección con la suma y multiplicación por 176 transformando las coordenadas en dirección

* El multiplexor no garantiza el intervalo de manera que si XPixel y YPixel estan fuera, apunta a la dirección 0

Amplificador Mapa

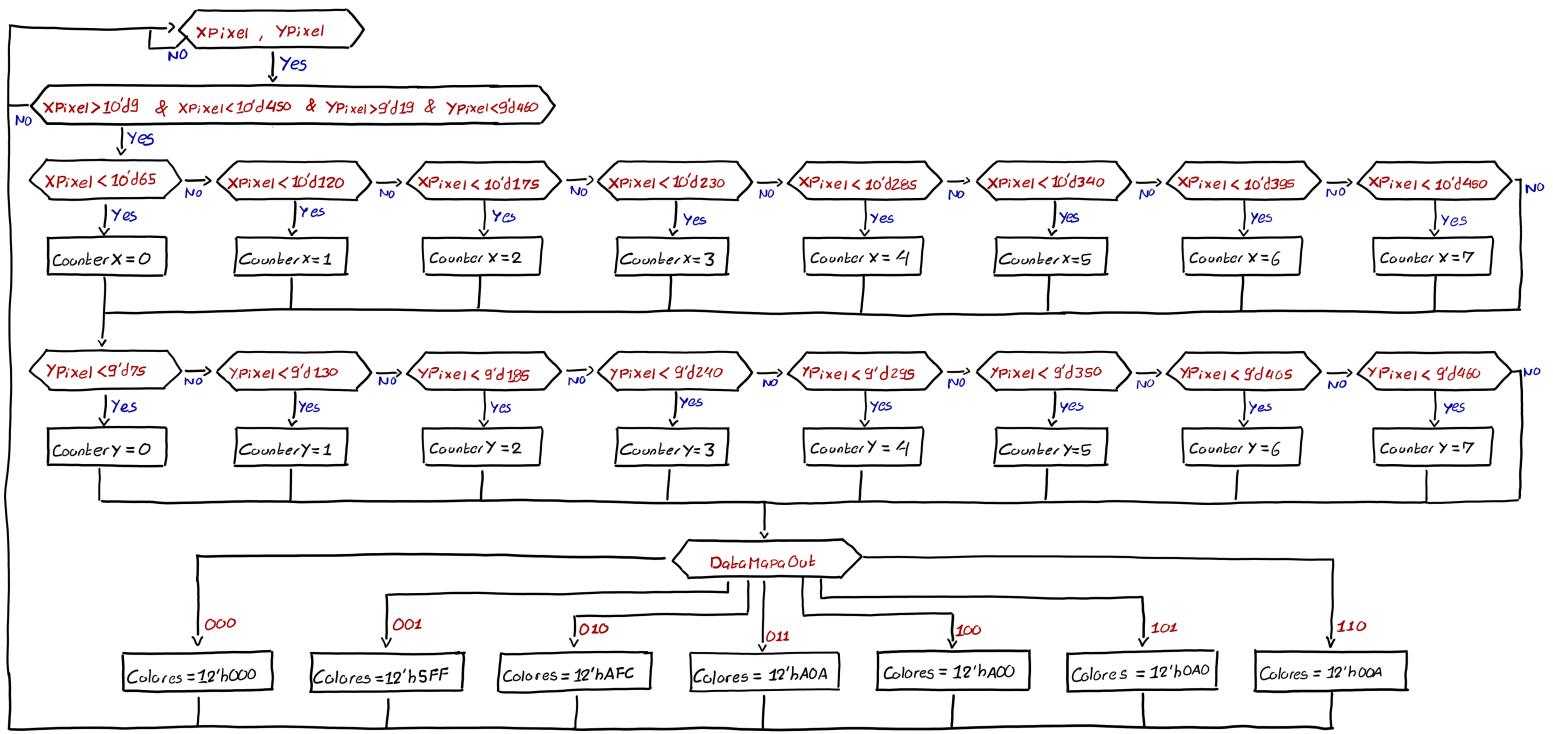
El mapa es un registro de 8×8 direcciones y 4 bits de profundidad, por lo cual resulta necesario decodificar los 4 bits de información en colores de 12 bits, ademas debido a las pocas direcciones normalmente se traduciría en 64 pixeles de la pantalla mostrando la información por lo cual son muy pocos respecto a los 307200 pixeles en total del protocolo usado, por ende se debe buscar la forma de retener la información amplificando el tamaño para que cada dirección del mapa se mueva en un mayor numero de pixeles



* Los cambios a los registros se sincronizan con latch de manera que por cada cambio en XPixel y YPixel se generan cambios

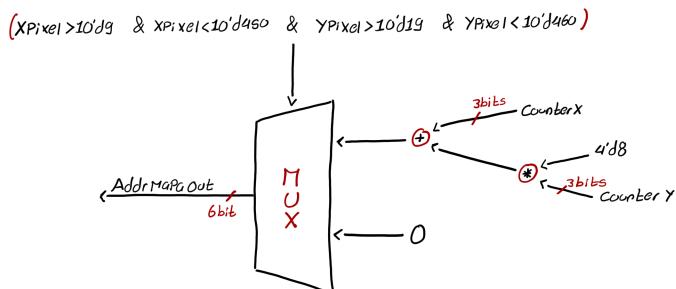
Proyecto Digital II → Camara/VGA/Driver

→ Diagrama Funcional



Gestor de direcciones del Mapa

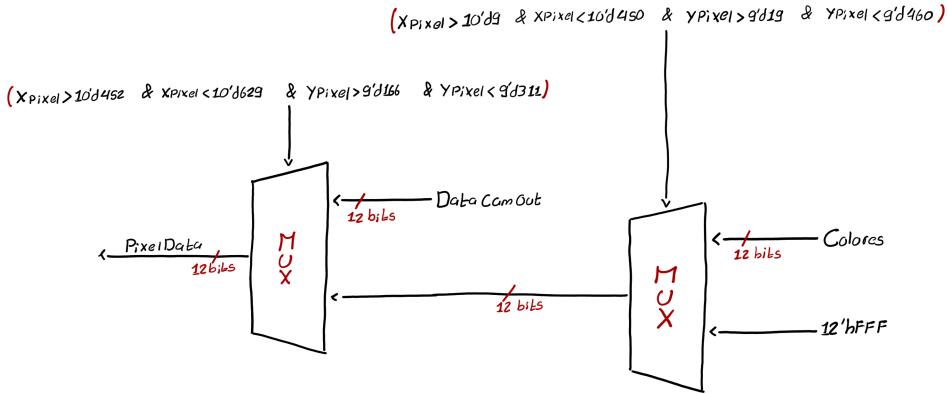
El gestor de direcciones del mapa funciona bajo la misma lógica del gestor de direcciones para la cámara. La región Asignada para el mapa es $10 < XPixel < 450$ y $20 < YPixel < 460$. Además no se deben hacer muchos ajustes pues CounterX y CounterY ya están adaptados.



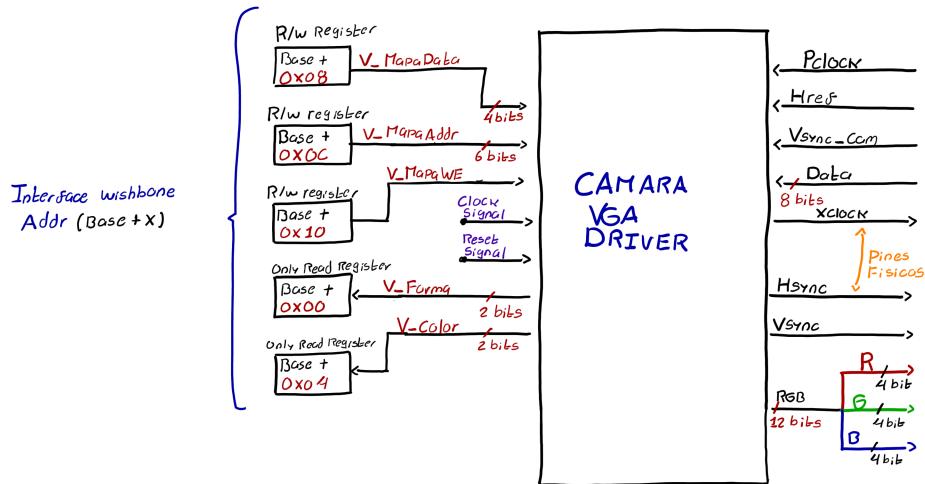
Proyecto Digital II → Camara/VGA/Driver

Gestor de espacio VGA

Una vez se tienen direcciones muy específicas para las direcciones de las memorias, se ha de ordenar la información de los colores asignando los mismos condicionales de manera que todo el espacio esté repartido de manera óptima



Modulo wishbone



Clocksignal hace referencia al reloj del sistema

Reset signal hace referencia a la señal de reinicio del sistema

Código C (Picorv 32)

→ La interacción con este módulo es muy sencilla debido a que casi todo funciona en hardware, por ende el siguiente código hace posible su funcionamiento

```
VGA_Draw (int xPos, int yPos, int Informacion)
    VGA_MAPA_MAPADATA_Write(Informacion); → Escribe el Color Actualizando el registro (Base + 0x08)
    VGA_MAPA_MAPA_Addr_Write(XPos + yPos * 8); → Escribe la dirección Actualizando el registro (Base + 0x0C)
    VGA_MAPA_MAPAwrite_Write(1); → Habilita la escritura del color en la casilla de la dirección Poniendo 1 en el registro (Base + 0x10)
    VGA_MAPA_MAPAwrite_Write(0); → Deshabilita la escritura Poniendo 0 en el registro (Base + 0x10)
}
```

La forma detectada se puede obtener leyendo el registro Forma (Base + 0x00), y en el caso del color se deberá leer el registro Color (Base + 0x04)