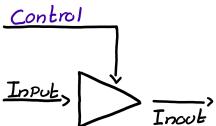


## Proyecto Digital → GPIO

### Modulo Basico (Sin Bus)



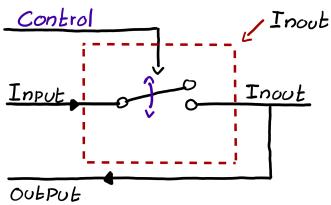
\* El modulo GPIO es Principalmente un Circuito Combinacional, Por lo que Se rige Por Una tabla de verdad

### → Tabla de estados

Control	Inout
1	Input
0	High-Z

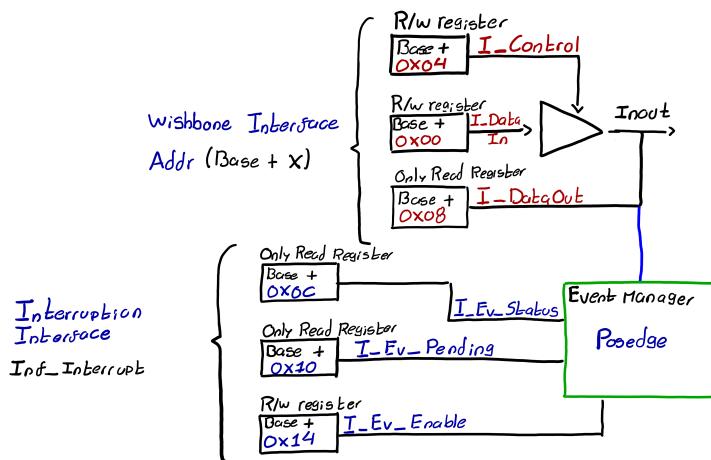
\* Teniendo en cuenta su funcionamiento, el diagrama de conexiones se puede ver de la siguiente manera

### → Diagrama de circuitos



\* Un Cable se conecta directamente a la salida de tal manera que cuando Control es 0, Output tomará lo que sea que entre en aquél Pin Inout, Actuando como un Input. Cuando Control es 1, el pin Inout tomará el valor de Input actuando como un Output y esto implica que Output tomará el valor de Input

### Modulo Wishbone (Incluye Interrupciones)



\* El tri-state no depende de un clock por lo cual no asignamos ClockSignal y ResetSignal

# Proyecto Digital → GPIO

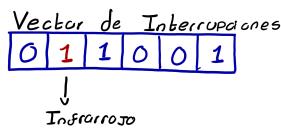
Código C (Picorv 32 - Interrupciones)

→ debido a que se tienen interrupciones se debe modificar `isr.c` agregando:

`if (irqs & (1 << Inf_Interrupt))` → Si la interrupción coincide con el vector de interrupción `Inf_Interrupt`

`infrarrojo_isr();`

Ejecuta la función



→ Las funciones para hacer posible el funcionamiento son:

`infrarrojo_init()`

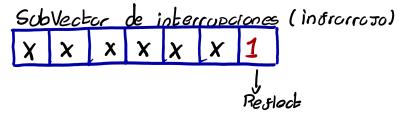
`Inf_ev_Pending_Write(Inf_ev_Pending_Read());`

`Inf_ev_Enable_Write(Inf_ev_Reflect);` → habilita las interrupciones de Reflect, del sub-vector de interrupciones `0x01`

`irq_SetMask(irq_GetMask() | (1 << Inf_Interrupt));`

habilita las interrupciones del infrarrojo en el vector de interrupciones general

}



`infrarrojo_disable(int disable)` { → Cuando no hay un objeto reflejante se obtiene un 1, cuando si refleja se obtiene un 0

`Inf_DataIn_Write(disable);` → Actualiza el registro (Base + `0x00`), si se deshabilita escribimos 1

`Inf_Control_Write(disable);` → Actualiza el registro (Base + `0x04`), si se deshabilita escribimos 1, de manera que el registro de DataOut se escribe con el valor de DataIn, por ende siempre se leerá 1

`infrarrojo_read()` {

} → Lee y retorna el valor del registro (Base + `0x08`), si 1 no hay reflejo, si 0 hay reflejo

`infrarrojo_isr()` {

`int stat;`

`irq_Setie(0);` → deshabilita todo interrupcio

`Stat = inf_ev_Pending_Read();` → Lee el subvector de interrupciones y lo almacena

`0x01`

`if (Stat & inf_ev_Reflect) {`

`Inf_ev_Pending_Write(Inf_ev_Reflect);`

`Uart_Write('X');` → Escribe X en consola

Se puede colocar en este segmento cualquier cosa que se desee con la interrupcion

}

`irq_Setie(1);` → habilita interrupciones

}