Puertos de entrada/salida digitales del LPC4088

Diseño Basado en Microprocesadores

Víctor Manuel Sánchez Corbacho

Dpto. de Automática, Electrónica, Arquitectura y Redes de Computadores

2017

Contenido

Puertos de entrada/salida digitales

- **Conjunto de pines** del encapsulado del microcontrolador que pueden usarse como **entradas o salidas digitales**.
- Desde los programas podemos **leer las entradas y modificar las salidas**.
- Aplicaciones:
  - Manejo de LEDs y otros indicadores.
  - Lectura del estado de interruptores y pulsadores.
  - Intercambio de datos con otros dispositivos.
  - Generación de pulsos digitales.

Aparatos con pulsadores y LEDs

- Mando de juegos.

- Balanza.

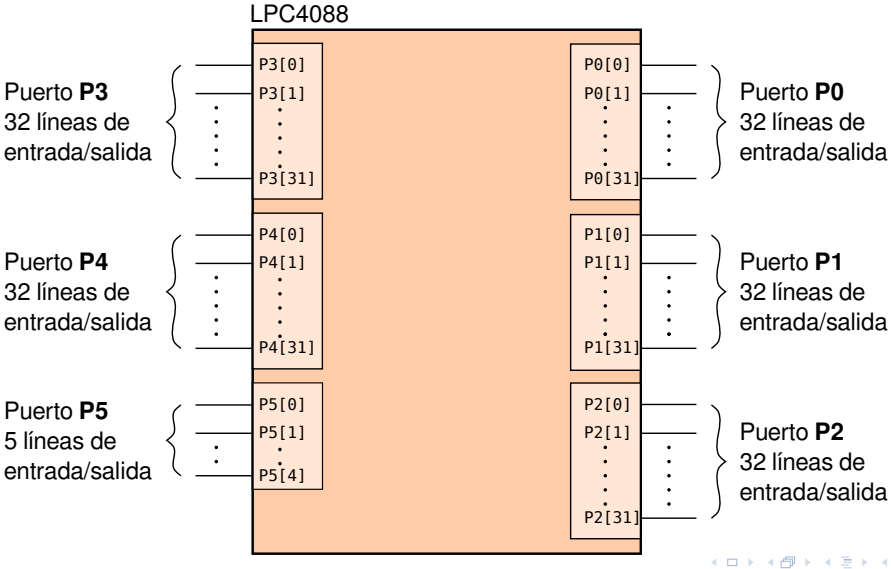
- Lavadora.

- Teclado de PC.

Puertos de E/S digitales del LPC4088

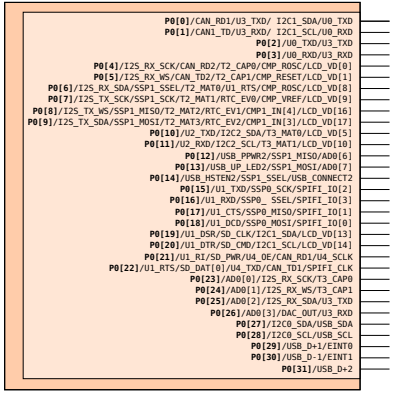
- Seis puertos en total: **P0 a P5**.
- Cada puerto consta de un **conjunto de pines de E/S**.
- Los pines de un puerto pueden usarse de forma conjunta o individualmente.
- Cada pin individual se nombra **P*número_puerto*[*número_pin*]**. Ejemplo: **P3[5]**
- Hasta 165 pines de E/S digital.
- Los puertos P0 y P2 pueden recibir señales de interrupción.
- Muchos pines tienen resistencias de pull-up y pull-down internas configurables.
- Manejo similar en todos los microcontroladores LPC.

Puertos de E/S digitales del LPC4088



Funciones alternativas de los pines de E/S digitales

- Cada pin de E/S puede asumir otras funciones alternativas.
- La función por defecto es pin de E/S digital.
- Más adelante se verá cómo seleccionar funciones alternativas.
- Ejemplo del puerto P0.



Registros de los puertos de E/S digitales

- **Cada puerto** se controla mediante un **conjunto de registros**.

Registro	**Función**
DIRx	Seleccionar la dirección entrada o salida de cada pin
PINx	Leer y escribir el estado de los pines del puerto
SETx	Poner uno o más pines del puerto a uno
CLRx	Poner uno o más pines del puerto a cero
MASKx	Enmascarar uno o más pines del puerto

x es el número del puerto.

(Más adelante veremos los registros de configuración IOCON, usados para configurar otras características de los pines, como funciones alternativas, pull-ups/pull-downs, filtros de entrada, slew-rate de salida, modo drenador abierto, modo analógico, etc.)

Todos los registros de puertos de E/S digitales (manual)

Table 93. Register overview: GPIO (base address 0x2009 8000)

Name	Access	Address offset	Description	Reset value	Table
DIR0	R/W	0x000	GPIO Port0 Direction control register.	0	95
MASK0	R/W	0x010	Mask register for Port0.	0	96
PIN0	R/W	0x014	Port0 Pin value register using FIOMASK.	0	97
SET0	R/W	0x018	Port0 Output Set register using FIOMASK.	0	98
CLR0	WO	0x01C	Port0 Output Clear register using FIOMASK.	-	99
DIR1	R/W	0x020	GPIO Port1 Direction control register.	0	95
MASK1	R/W	0x030	Mask register for Port1.	0	96
PIN1	R/W	0x034	Port1 Pin value register using FIOMASK.	0	97
SET1	R/W	0x038	Port1 Output Set register using FIOMASK.	0	98
CLR1	WO	0x03C	Port1 Output Clear register using FIOMASK.	-	99
DIR2	R/W	0x040	GPIO Port2 Direction control register.	0	95
MASK2	R/W	0x050	Mask register for Port2.	0	96
PIN2	R/W	0x054	Port2 Pin value register using FIOMASK.	0	97
SET2	R/W	0x058	Port2 Output Set register using FIOMASK.	0	98
CLR2	WO	0x05C	Port2 Output Clear register using FIOMASK.	-	99
DIR3	R/W	0x060	GPIO Port3 Direction control register.	0	95
MASK3	R/W	0x070	Mask register for Port3.	0	96
PIN3	R/W	0x074	Port3 Pin value register using FIOMASK.	0	97
SET3	R/W	0x078	Port3 Output Set register using FIOMASK.	0	98
CLR3	WO	0x07C	Port3 Output Clear register using FIOMASK.	-	99
DIR4	R/W	0x080	GPIO Port4 Direction control register.	0	95
MASK4	R/W	0x090	Mask register for Port4.	0	96
PIN4	R/W	0x094	Port4 Pin value register using FIOMASK.	0	97
SET4	R/W	0x098	Port4 Output Set register using FIOMASK.	0	98
CLR4	WO	0x09C	Port4 Output Clear register using FIOMASK.	-	99
DIR5	R/W	0x0A0	GPIO Port5 Direction control register.	0	95
MASK5	R/W	0x0B0	Mask register for Port5.	0	96
PIN5	R/W	0x0B4	Port5 Pin value register using FIOMASK.	0	97
SET5	R/W	0x0B8	Port5 Output Set register using FIOMASK.	0	98
CLR5	WO	0x0BC	Port5 Output Clear register using FIOMASK.	-	99

Relación entre pines y bits de los registros del puerto

- **Cada pin** de entrada/salida de cada puerto **depende de**:
  - Un bit en el registro DIR del puerto.
  - Un bit en el registro PIN del puerto.
  - Un bit en el registro SET del puerto.
  - Un bit en el registro CLR del puerto.
  - Un bit en el registro MASK del puerto.

Ejemplo

- El pin **P3[5] depende de**:
  - El **bit 5** del registro **DIR3** para especificar su dirección.
  - El **bit 5** del registro **PIN3** para leer su estado si es entrada o modificar su estado si es salida.
  - El **bit 5** del registro **SET3** para ponerlo a 1.
  - El **bit 5** del registro **CLR3** para ponerlo a 0.
  - El **bit 5** del registro **MASK3** para enmascararlo.

Registros de dirección DIRx

- Permiten **configurar cada pin** de un puerto **como entrada o como salida**.
- Pueden escribirse para configurar la dirección de cada pin de E/S digital.
  - Un bit a **0** configura el pin correspondiente como **entrada**.
  - Un bit a **1** configura el pin correspondiente como **salida**.
- Pueden leerse para conocer la configuración de dirección actual de los pines.

Ejemplo

- Al *resetear* el microcontrolador todos los registros DIR se ponen a cero ⇒
  - **Al *resetear* el microcontrolador todos los pines se configuran como entrada**.

Registros de estado de pines PINx

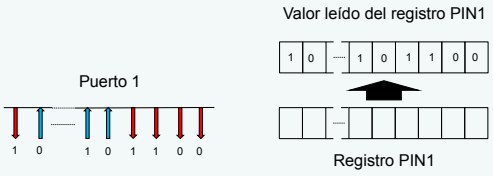
- Permiten modificar el estado de los pines configurados como salidas y conocer el estado externo de los pines (tanto los configurados como entradas como los configurados como salidas.)
- **Escritura en un registro PINx**: todos los pines del puerto x configurados como salida cambian al estado del correspondiente bit en el dato escrito. No afecta a las entradas.

Ejemplo

Registros de estado de pines PINx

- **Lectura de un registro PINx**: se **lee el estado externo** de todos los pines del puerto x, **tanto entradas como salidas** [1].

Ejemplo

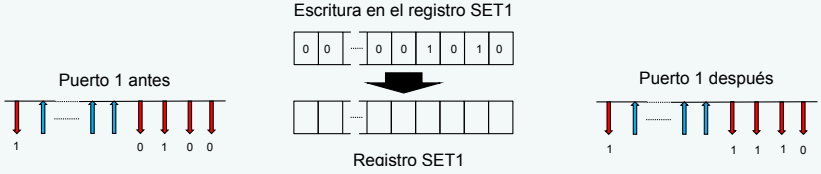


[1] Puede darse el caso de que estado externo de un pin configurado como salida no coincida con el estado fijado por el programa (mediante escrituras en los registros PINx, SETx y CLRx) si el pin está siendo forzando externamente a un nivel distinto. Esto es generalmente una condición anómala para un pin configurado con etapa de salida push-pull pero normal si la etapa de salida del pin está configurada en drenador abierto.

Registros de puesta a uno SETx

- Facilita **poner a 1 uno o más pines de salida sin afectar al resto**.
- Al escribir en este registro:
  - **Sólo afecta** a los pines que están configurados como **salidas**.
  - Los **bits a 1 en el dato escrito ponen a 1** el correspondiente pin de salida.
  - Los **bits a 0 en el dato escrito no afectan** al correspondiente pin de salida.

Ejemplo

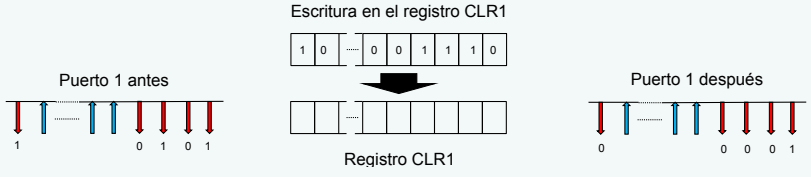


- Al leer este registro se obtiene el valor determinado por las escrituras previas en SETx, CLRx y PINx **sin la influencia de cualquier acción externa sobre el pin**.

Registros de puesta a cero CLRx

- Facilita **poner a 0 uno o más pines de salida sin afectar al resto**.
- Al escribir en este registro.
  - **Sólo afecta** a los pines que están configurados como **salidas**.
  - Los **bits a 1 en el dato escrito ponen a 0** el correspondiente pin de salida.
  - Los **bits a 0 en el dato escrito no afectan** al correspondiente pin de salida.

Ejemplo



Registros de máscara MASKx

- Permiten **proteger de cambios los pines de salida y enmascarar el estado de los pines de entrada**.
- Si el bit correspondiente a un pin en el registro MASK asociado está a 0:
  - El pin opera normalmente.
- Si el bit correspondiente a un pin en el registro MASK asociado está a 1:
  - Las escrituras en los registros PIN, SET y CLR asociado a pin no le afectan.
  - Las lecturas del registro PIN siempre devuelven el correspondiente bit a 0.
- Al *resetear* el microcontrolador todos los registros MASK se ponen a 0 $\Rightarrow$
  - **Al *resetear* el microcontrolador todos los pines operan normalmente**.

Acceso a los registros desde C (I)

- **Incluir el fichero `LPC407x_8x_177x_8x.h`**
- Define estructuras que corresponden a los bloques de registros de periféricos.

Extracto del fichero `LPC407x_8x_177x_8x.h`

```
/*--------- General Purpose Input/Output (GPIO) ----------------------*/
typedef struct
{
  __IO uint32_t DIR;
       uint32_t RESERVED0[3];
  __IO uint32_t MASK;
  __IO uint32_t PIN;
  __IO uint32_t SET;
  __O  uint32_t CLR;
} LPC_GPIO_TypeDef;
```

```
typedef unsigned int uint32_t; /* Definido en stdint.h   */
#define __I   volatile const   /* Definido en core_cm4.h */
#define __O   volatile         /*          "             */
#define __IO  volatile         /*          "             */
```

Acceso a los registros desde C (II)

- `LPC407x_8x_177x_8x.h` también define punteros a estructuras mapeadas sobre las direcciones de los bloques de registros.

Extracto del fichero `LPC407x_8x_177x_8x.h`

```
#define LPC_AHB_BASE          (0x20080000UL)
...
#define LPC_GPIO0_BASE        (LPC_AHB_BASE  + 0x18000)
#define LPC_GPIO1_BASE        (LPC_AHB_BASE  + 0x18020)
#define LPC_GPIO2_BASE        (LPC_AHB_BASE  + 0x18040)
#define LPC_GPIO3_BASE        (LPC_AHB_BASE  + 0x18060)
#define LPC_GPIO4_BASE        (LPC_AHB_BASE  + 0x18080)
#define LPC_GPIO5_BASE        (LPC_AHB_BASE  + 0x180A0)
...
#define LPC_GPIO0             ((LPC_GPIO_TypeDef*)LPC_GPIO0_BASE)
#define LPC_GPIO1             ((LPC_GPIO_TypeDef*)LPC_GPIO1_BASE)
#define LPC_GPIO2             ((LPC_GPIO_TypeDef*)LPC_GPIO2_BASE)
#define LPC_GPIO3             ((LPC_GPIO_TypeDef*)LPC_GPIO3_BASE)
#define LPC_GPIO4             ((LPC_GPIO_TypeDef*)LPC_GPIO4_BASE)
#define LPC_GPIO5             ((LPC_GPIO_TypeDef*)LPC_GPIO5_BASE)
```

Acceso a los registros desde C (III)

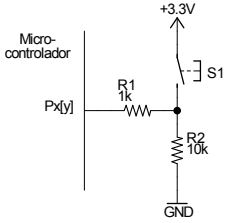
- Para acceder a un registro se escribe:
  - El nombre del puntero al bloque de registros `LPC_GPIOx`, siendo x el número del puerto.
  - Seguidamente el operador acceso a miembro `->`
  - A continuación, el nombre del registro sin el número de puerto.

Ejemplos

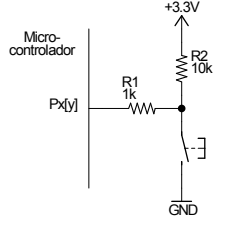
```
LPC_GPIO1->DIR = 6;     /* Escritura en DIR1 */

v = LPC_GPIO4->PIN;     /* Lectura de PIN4 */
```

Conexión de pulsadores

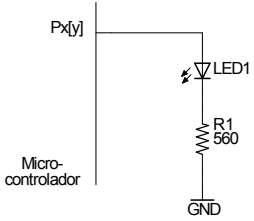


- Con esta conexión, cuando en el programa leamos el pin:
  - Encontraremos el pin 1 mientras el pulsador esté pulsado.
  - Encontraremos el pin 0 mientras el pulsador no esté pulsado.
- R2 establece un nivel eléctrico bajo en el pin mientras el pulsador no está pulsado. Por ello se llama resistencia de *pull-down*.
- R1 protege al pin si por error se configura como salida y se escribe a 0.

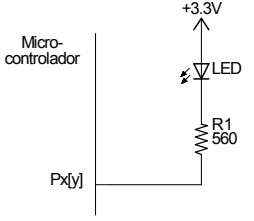


- Con esta conexión, cuando en el programa leamos el pin:
  - Encontraremos el pin 0 mientras el pulsador esté pulsado.
  - Encontraremos el pin 1 mientras el pulsador no esté pulsado.
- R2 establece un nivel eléctrico alto en el pin mientras el pulsador no está pulsado. Por ello se llama resistencia de *pull-up*.
- R1 protege al pin si por error se configura como salida y se escribe a 1.

Conexión de LEDs



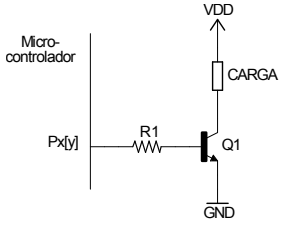
- Con esta conexión, cuando en el programa escribamos en el pin:
  - El LED se encenderá al escribir un 1.
  - El LED se apagará al escribir un 0.



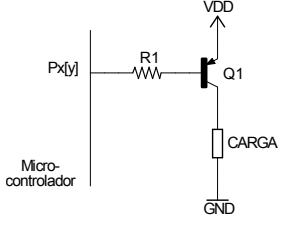
- Con esta conexión, cuando en el programa escribamos en el pin:
  - El LED se encenderá al escribir un 0.
  - El LED se apagará al escribir un 1.

Conexión de cargas más grandes mediante transistor

- Pin a 1 enciende. Pin a 0 apaga.

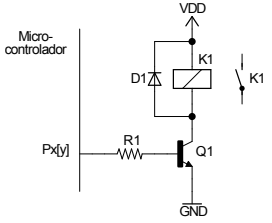


- Pin a 0 enciende. Pin a 1 apaga.



Conexión a relé

- Pin a 1 activa el relé. Pin a 0 lo desactiva.



- Pin a 0 activa el relé. Pin a 1 lo desactiva.