Guía mínima de Puertos de entrada salida TM4C1294

Nota: Todos los GPIO son tolerantes a 5-V cuando se configuran como entradas, excepto para PB0 y PB1, que están limitados a 3.6V

La arquitectura ARM permite un esquema de "lee-modifica-almacena" (**READ-MODIFY-STORE**), de tal forma que la modificación de los valores de los registros se puede hacer de esta manera, especialmente cuando un registro agrupa la configuración de diversos módulos, con el fin de modificar solo los bits del módulo de interés sin afectar los demás.

Los pasos recomendados para programar un GPIO como Digital general (sin protocolo en particular)

1. **Activar el Reloj** para el puerto en cuestión en el registro **RCGCGPIO** del Sistema de Control. Esperar a que se estabilice el modulo; en el registro **PRGPIO** (General-Purpose Input/Output **Peripheral Ready**) del Sistema de Control, se activa el bit del puerto correspondiente que indica que el puerto está listo para acceso.
2. **Configurar la dirección** (Input/Output) de las terminales del puerto con el registro **GPIODIR**.

0 Entrada	1 Salida

3. Programar los bits del registro **GPIOAFSEL** para emplearlos como **GPIO digitales**, o para emplearlos con una Función Alterna. (En Reset, para la mayoría de los pines, GPIOAFSEL = 0)

0 GPIO Digital	1 Función Alterna

4. Programar la función alterna (no corresponde a esta práctica).
5. Por default las terminales entregan 2mA. Para una configuración diferente, se programan los bits correspondientes del registro GPIOPC, así como los registros GPIODRxR.
6. Programar los registros de Pullup / Pulldown / Opendrain **GPIOPUR, GPIOPDR, GPIOODR**.
7. Habilitar el modo Digital con el registro **GPIODEN**.
8. Configurar parámetros para interrupción (no corresponde a esta práctica).

Para acceder a los bits específicos, se emplea la dirección Base del Puerto y se le suma el valor de cada bit según la siguiente tabla.

If we wish to access bit	*Constant*
7	0x0200
6	0x0100
5	0x0080
4	0x0040
3	0x0020
2	0x0010
1	0x0008
0	0x0004

Por ejemplo, si se quiere acceder a todos los bits a la vez, se lee todo el puerto definiendo la dirección:

Dirección base del Puerto + 0x3FC

A.S.H.D.

Creación del archivo fuente para emplear los puertos GPIO

Para tener acceso a los registros de configuración, se requiere conocer sus direcciones en el mapa de memoria.
Una forma de conocer estas direcciones es extrayéndolas del manual del microprocesador, lo cual a demás de tomar tiempo es bastante tedioso.
La otra forma es aprovechar que alguien más ya hizo el trabajo, definiendo cada dirección y un nombre simbólico que podemos usar dentro del programa.
Estos nombres y direcciones están en un pequeño archivo llamado *"gpio_regs.s"* que contiene solo los puertos GPIO y del Sistema de control.

Por ejemplo, el registro que contiene el bit que controla la habilitación del Reloj para los puertos GPIO consiste en la localidad de memoria con la siguiente dirección.

SYSCTL_RCGCGPIO_R .equ 0x400FE608
Y el registro donde leemos el estado de las terminales es en el registro con la siguiente dirección.
GPIO_PORTA_DATA_R .equ 0x400583FC

Una forma de agregar esta información al programa es definiendo en memoria ROM cada una de estas direcciones y asociarla con un símbolo de la siguiente forma

SYSCTL_RCGCGPIO_R .field 0x400FE608
GPIO_PORTA_DATA_R .field 0x400583FC

Lo cual ocupa espacio de memoria ROM. Una forma de hacer uso eficiente de la memoria ROM es colocar el valor de la dirección solo donde se requiera, haciendo referencia con el símbolo asociado. Para esto usaremos el archivo gpio_regs.s y una Macro que realiza la carga del valor de la dirección de 32 bits en un registro de trabajo haciendo dos cargas de 16 bits, en primer lugar cargando la parte baja de la dirección en el registro y después cargando la parte alta en los 16 bits más altos.

La carga de un valor en un registro emplea direccionamiento inmediato. Este modo de direccionamiento es explícito en el cuerpo de la Macro, pero no es explicito al usarla. A continuación se muestra la Macro:

```
LDR32 .macro P1, P2                          ; P1, P2 son argumentos de la macro
                                           ; P1 es el Destino, P2 es la Fuente
      .eval  (P2 & 0xFFFF0000) >> 16, temp     ; el símbolo temp contiene los 16 MSB
      .eval  P2 & 0x0000FFFF, P2               ; P2 contiene los 16 LSB
       MOVW  P1, #P2                           ; carga en destino 16 LSB
        MOVT  P1, #temp                        ; carga en destino 16 MSB
      .endm                                    ; fin de macro
```

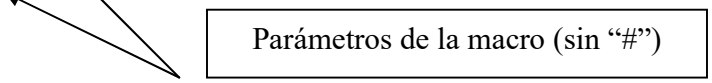
Direccionamiento inmediato

A.S.H.D.

Al usar la Macro, el direccionamiento inmediato no es explicito como en las instrucciones nativas, por lo que hay que tener cuidado al emplear la Macro:

```
val   .equ  0x12345678  ; valor definido en el mismo archivo fuente
                        ; o en un archivo aparte

          LDR32 R0, val   ; carga del dato 0x12345678 en R0 empleando la Macro
          LDR32 R1, 0xFFFF0000
```



Programa fuente básico. Encendido y Apagado de un Bit.

A continuación se describen las partes del programa.
Archivos a incluir. Con la directiva .include se especifican los archivos en donde se tiene la lista de los nombres y direcciones de los registros GPIO y el archivo donde se define la macro.

```
.include "gpio_reg.s"
.include "macros.s"
```

La estructura del programa es la siguiente: Llama a una subrutina de inicialización de puertos PortF_Init, llama a una función llamada led_toggle para cambiar el estado de un LED, llama a una función de retardo, se repite lo mismo haciendo salto incondicional a la dirección Loop1. Notar que se hace uso de la instrucción **BL para llamado a subrutinas y BX LR para regreso de éstas.**

```
                .global main
                .text
main:
                BL  PortF_Init
Loop1           BL  led_toggle
                BL  delay
                B   Loop1
```

Las funciones.

PortF_Init: Se hace uso de la Macro para cargar la dirección de 32 bits en un Registro para usarlo como apuntador ①, extraer su contenido ② y modificarlo③, o escribir directamente un nuevo valor ④.

```
PortF_Init:
                LDR32 R1, SYSCTL_RCGCGPIO_R     ; 1) ① activate clock for Port F
                LDR   R0, [R1]                    ②
                ORR   R0, R0, #0x20             ;  ③ set bit 5 to turn on clock
                STR   R0, [R1]

                LDR32 R1, GPIO_PORTF_DIR_R      ; 2) set direction register
                MOV   R0, #0x01                 ;  ④ PF0 output
                STR   R0, [R1]

                LDR32 R1, GPIO_PORTF_DEN_R      ; 3) enable Port F digital port
                MOV   R0, #0x1                          ;  1 means enable digital I/O
                STR   R0, [R1]
                BX    LR
```

La función led_toggle: Se apunta al registro de datos del Puerto, se extrae su contenido, se modifica y se escribe el valor resultante de regreso en el registro de datos del puerto.

```
led_toggle:
          LDR32 R1, GPIO_PORTF_DATA_R     ; (1) pointer to Port F Data Register
          LDR   R0, [R1]                  ; (2) read all of Port F
          EOR   R0, R0, #0x01             ; (3) NOT a bit 0, guarda en R0
          ;LDR32 R1, GPIO_PORTF_DATA_R    ; (4) escribe en el reg de datos
          STR   R0, [R1]
          BX    LR
```

Función de retardo.

Su función solo es consumir ciclos de reloj. Entre cada cambio de estado de la terminal donde conectamos el LED, el programa se quedará en ese estado hasta que se termine el retardo y vuelva a cambiar de estado, con la finalidad de que podamos apreciar a simple vista ese cambio. Se implementa un contador con un ciclo, el valor del contador dependerá del tiempo que queremos que se consuma. El reloj del sistema es de 16 MHz después de un Reset. Empleando el periodo de un reloj de 16 MHz (62.5 [ns]), se calcula de forma aproximada el tiempo, por ejemplo para 100 ms.

Según las especificaciones, las instrucciones tienen la siguiente duración en ciclos de reloj:

Operación	Descripción	Ensamblador	Ciclos
Subtract	Subtract	SUB Rd, Rn, <op>	1
Branch	Condicional	B<cc> <label>	1 or 1+P[c]

donde

P: El numero de ciclos requeridos para un llenado completo del ciclo de pipeline, que va de 1 a 3 dependiendo de la alineación y la longitud de la instrucción objetivo.

c: El salto (branch) condicional se completa en un ciclo si no se realiza el salto.

El ciclo de llenado del pipe line será de 3 ciclos requeridos para: Fetch, Decode, Execute.

```
delay:
                LDR32 R8, 0x2FFFF     ; CONTADOR de N ciclos: 196,607 * T_clock =
dl_loop         SUBS  R8, #1          ; 1 ciclo
                BNE   dl_loop         ; 3 ciclos cuando si realiza el salto, 1 cuando no
                BX    LR
```

Resulta que el delay del ejemplo, implementa un tiempo de

$$delay = [(1\ cy_{SUB} + 3\ cy_{BNE})(N - 1) + (1\ cy_{SUB} + 1 cy_{BNE})] * T_{clock}$$

$$delay = [4(196606) + 2] * 62.5\ ns = 0.049151625\ [s] \approx 50[ms]$$

De este resultado se concluye que cuando la terminal se encuentra en estado alto, permanece durante 50 ms aproximadamente; cuando cambia a estado bajo, permanece durante otros 50 ms aproximadamente, por lo que un ciclo completo “alto-bajo” tiene una duración de 100 [ms], dando forma a una señal con frecuencia de 10 Hz, que es posible apreciar a simple vista. Esta frecuencia de 10 Hz corresponde a la frecuencia de cambio de estado, y no como se podría pensar en un primer inicio, que el LED encendería 10 veces por segundo. Para este caso, la frecuencia debería ser del doble (20 Hz), así tendríamos 10 estados encendidos y 10 estados apagados en el periodo de un segundo.

Tabla 10-7 Mapa de registros GPIO

Offset	Name	Type	Reset	Description	See page
0x000	GPIODATA	RW	0x0000.0000	GPIO Data	759
0x400	GPIODIR	RW	0x0000.0000	GPIO Direction	760
0x404	GPIOIS	RW	0x0000.0000	GPIO Interrupt Sense	761
0x408	GPIOIBE	RW	0x0000.0000	GPIO Interrupt Both Edges	762
0x40C	GPIOIEV	RW	0x0000.0000	GPIO Interrupt Event	763
0x410	GPIOIM	RW	0x0000.0000	GPIO Interrupt Mask	764
0x414	GPIORIS	RO	0x0000.0000	GPIO Raw Interrupt Status	765
0x418	GPIOMIS	RO	0x0000.0000	GPIO Masked Interrupt Status	767
0x41C	GPIOICR	W1C	0x0000.0000	GPIO Interrupt Clear	769
0x420	GPIOAFSEL	RW	-	GPIO Alternate Function Select	770
0x500	GPIODR2R	RW	0x0000.00FF	GPIO 2-mA Drive Select	772
0x504	GPIODR4R	RW	0x0000.0000	GPIO 4-mA Drive Select	773
0x508	GPIODR8R	RW	0x0000.0000	GPIO 8-mA Drive Select	774
0x50C	GPIOODR	RW	0x0000.0000	GPIO Open Drain Select	775
0x510	GPIOPUR	RW	-	GPIO Pull-Up Select	776
0x514	GPIOPDR	RW	0x0000.0000	GPIO Pull-Down Select	778
0x518	GPIOSLR	RW	0x0000.0000	GPIO Slew Rate Control Select	780
0x51C	GPIODEN	RW	-	GPIO Digital Enable	781
0x520	GPIOLOCK	RW	0x0000.0001	GPIO Lock	783
0x524	GPIOCR	-	-	GPIO Commit	784
0x528	GPIOAMSEL	RW	0x0000.0000	GPIO Analog Mode Select	786
0x52C	GPIOPCTL	RW	-	GPIO Port Control	787
0x530	GPIOADCCTL	RW	0x0000.0000	GPIO ADC Control	789
0x534	GPIODMACTL	RW	0x0000.0000	GPIO DMA Control	790
0x538	GPIOSI	RW	0x0000.0000	GPIO Select Interrupt	791
0x53C	GPIODR12R	RW	0x0000.0000	GPIO 12-mA Drive Select	792
0x540	GPIOWAKEPEN	RW	0x0000.0000	GPIO Wake Pin Enable	793
0x544	GPIOWAKELVL	RW	0x0000.0000	GPIO Wake Level	795
0x548	GPIOWAKESTAT	RO	0x0000.0000	GPIO Wake Status	797
0xFC0	GPIOPP	RO	0x0000.0001	GPIO Peripheral Property	799
0xFC4	GPIOPC	RW	0x0000.0000	GPIO Peripheral Configuration	800
0xFD0	GPIOPeriphID4	RO	0x0000.0000	GPIO Peripheral Identification 4	803
0xFD4	GPIOPeriphID5	RO	0x0000.0000	GPIO Peripheral Identification 5	804
0xFD8	GPIOPeriphID6	RO	0x0000.0000	GPIO Peripheral Identification 6	805
0xFDC	GPIOPeriphID7	RO	0x0000.0000	GPIO Peripheral Identification 7	806
0xFE0	GPIOPeriphID0	RO	0x0000.0061	GPIO Peripheral Identification 0	807
0xFE4	GPIOPeriphID1	RO	0x0000.0000	GPIO Peripheral Identification 1	808
0xFE8	GPIOPeriphID2	RO	0x0000.0018	GPIO Peripheral Identification 2	809
0xFEC	GPIOPeriphID3	RO	0x0000.0001	GPIO Peripheral Identification 3	810
0xFF0	GPIOPCellID0	RO	0x0000.000D	GPIO PrimeCell Identification 0	811
0xFF4	GPIOPCellID1	RO	0x0000.00F0	GPIO PrimeCell Identification 1	812
0xFF8	GPIOPCellID2	RO	0x0000.0005	GPIO PrimeCell Identification 2	813
0xFFC	GPIOPCellID3	RO	0x0000.00B1	GPIO PrimeCell Identification 3	814

A.S.H.D.

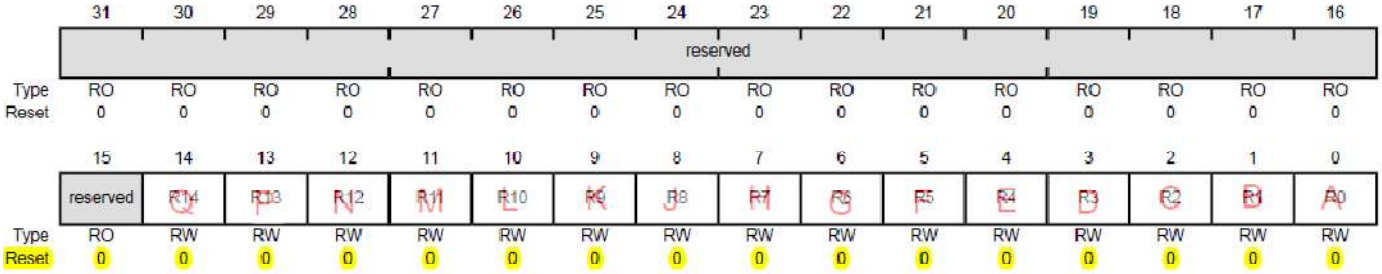
Register 89: General-Purpose Input/Output Run Mode Clock Gating Control (RCGCGPIO), offset 0x608

The **RCGCGPIO** register provides software the capability to enable and disable GPIO modules in Run mode. When enabled, a module is provided a clock and accesses to module registers are allowed. When disabled, the clock is disabled to save power and accesses to module registers generate a bus fault.

Important: This register should be used to control the clocking for the GPIO modules.

General-Purpose Input/Output Run Mode Clock Gating Control (RCGCGPIO)

Base 0x400F.E000
Offset 0x608
Type RW, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:15	reserved	RO	0	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
14	R14	RW	0	GPIO Port Q Run Mode Clock Gating Control Value Description 0 GPIO Port Q is disabled. 1 Enable and provide a clock to GPIO Port Q in Run mode.

Register 1: GPIO Data (GPIODATA), offset 0x000

The **GPIODATA** register is the data register. In software control mode, values written in the **GPIODATA** register are transferred onto the GPIO port pins if the respective pins have been configured as outputs through the **GPIO Direction (GPIODIR)** register (see page 760).

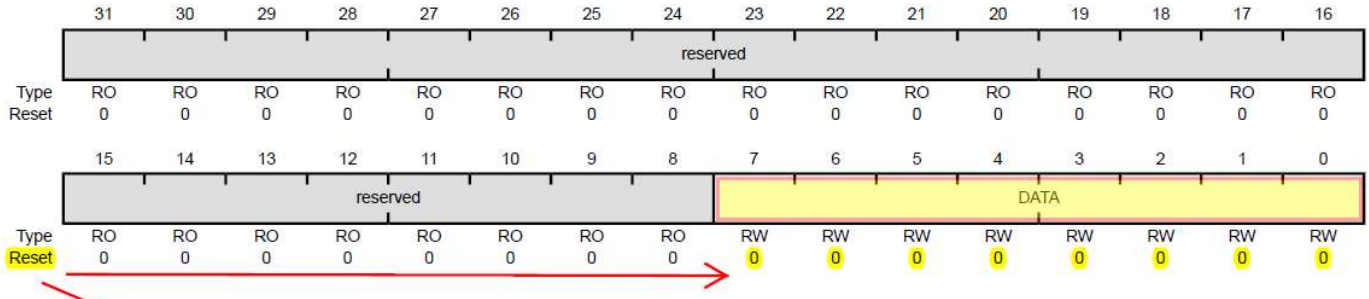
In order to write to **GPIODATA**, the corresponding bits in the mask, resulting from the address bus bits [9:2], must be set. Otherwise, the bit values remain unchanged by the write.

Similarly, the values read from this register are determined for each bit by the mask bit derived from the address used to access the data register, bits [9:2]. Bits that are set in the address mask cause the corresponding bits in **GPIODATA** to be read, and bits that are clear in the address mask cause the corresponding bits in **GPIODATA** to be read as 0, regardless of their value.

A read from **GPIODATA** returns the last bit value written if the respective pins are configured as outputs, or it returns the value on the corresponding input pin when these are configured as inputs. All bits are cleared by a reset.

GPIO Data (GPIODATA)

GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (AHB) base: 0x4006.0000
GPIO Port K (AHB) base: 0x4006.1000
GPIO Port L (AHB) base: 0x4006.2000
GPIO Port M (AHB) base: 0x4006.3000
GPIO Port N (AHB) base: 0x4006.4000
GPIO Port P (AHB) base: 0x4006.5000
GPIO Port Q (AHB) base: 0x4006.6000
Offset 0x000
Type RW, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	RW	0x00	GPIO Data This register is virtually mapped to 256 locations in the address space. To facilitate the reading and writing of data to these registers by independent drivers, the data read from and written to the registers are masked by the eight address lines [9:2]. Reads from this register return its current state. Writes to this register only affect bits that are not masked by ADDR[9:2] and are configured as outputs. See “Data Register Operation” on page 749 for examples of reads and writes.

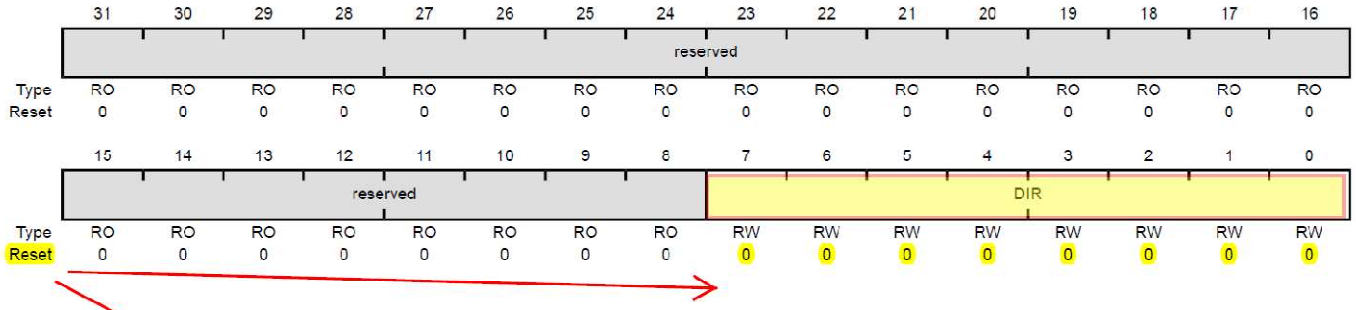
A.S.H.D.

Register 2: GPIO Direction (GPIODIR), offset 0x400

The **GPIODIR** register is the data direction register. Setting a bit in the **GPIODIR** register configures the corresponding pin to be an output, while clearing a bit configures the corresponding pin to be an input. All bits are cleared by a reset, meaning all GPIO pins are inputs by default.

GPIO Direction (GPIODIR)

GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (AHB) base: 0x4006.0000
GPIO Port K (AHB) base: 0x4006.1000
GPIO Port L (AHB) base: 0x4006.2000
GPIO Port M (AHB) base: 0x4006.3000
GPIO Port N (AHB) base: 0x4006.4000
GPIO Port P (AHB) base: 0x4006.5000
GPIO Port Q (AHB) base: 0x4006.6000
Offset 0x400
Type RW, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DIR	RW	0x00	GPIO Data Direction

Value	Description
0	Corresponding pin is an input.
1	Corresponding pins is an output.

Register 10: GPIO Alternate Function Select (GPIOAFSEL), offset 0x420

Note: Tamper pins enabled in the **Hibernate Tamper IO Control and Status (HIBTPIO)** register override the AFSEL configuration.

The **GPIOAFSEL** register is the mode control select register. If a bit is clear, the pin is used as a GPIO and is controlled by the GPIO registers. Setting a bit in this register configures the corresponding GPIO line to be controlled by an associated peripheral. Several possible peripheral functions are multiplexed on each GPIO. The **GPIO Port Control (GPIOPCTL)** register is used to select one of the possible functions. Table 26-5 on page 1817 details which functions are muxed on each GPIO pin. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in the table below.

Important: The table below shows special consideration GPIO pins. Most GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0). Special consideration pins may be programed to a non-GPIO function or may have special commit controls out of reset. In addition, a Power-On-Reset ($\overline{\text{POR}}$) returns these GPIO to their original special consideration state.

Table 10-8. GPIO Pins With Special Considerations

GPIO Pins	Default Reset State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL	GPIOCR
PC[3:0]	JTAG/SWD	1	1	0	1	0x1	0
PD[7]	GPIO[a]	0	0	0	0	0x0	0
PE[7]	GPIO[a]	0	0	0	0	0x0	0

a. This pin is configured as a GPIO by default but is locked and can only be reprogrammed by unlocking the pin in the **GPIOLOCK** register and uncommitting it by setting the **GPIOCR** register.

The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware signals including the GPIO pins that can function as JTAG/SWD signals and the NMI signal. The commit control process must be followed for these pins, even if they are programmed as alternate functions other than JTAG/SWD or NMI; see "Commit Control" on page 752.

Note: If the device fails initialization during reset, the hardware toggles the TDO output as an indication of failure. Thus, during board layout, designers should not designate the TDO pin as a GPIO in sensitive applications where the possibility of toggling could affect the design.

Caution – It is possible to create a software sequence that prevents the debugger from connecting to the TM4C1294NCPDT microcontroller. If the program code loaded into flash immediately changes the JTAG pins to their GPIO functionality, the debugger may not have enough time to connect and halt the controller before the JTAG pin functionality switches. As a result, the debugger may be locked out of the part. This issue can be avoided with a software routine that restores JTAG functionality based on an external or software trigger. In the case that the software routine is not implemented and the device is locked out of the part, this issue can be solved by using the TM4C1294NCPDT Flash Programmer "Unlock" feature. Please refer to LMFLASHPROGRAMMER on the TI web for more information.

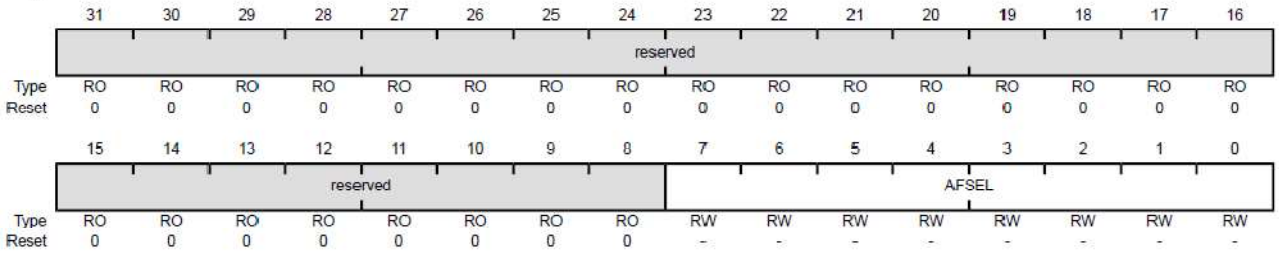
A.S.H.D.

The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the GPIO pins that can be used as the four JTAG/SWD pins and the NMI pin (see "Signal Tables" on page 1781 for pin numbers). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 770), **GPIO Pull Up Select (GPIOPUR)** register (see page 776), **GPIO Pull-Down Select (GPIOPDR)** register (see page 778), and **GPIO Digital Enable (GPIODEN)** register (see page 781) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 783) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 784) have been set.

When using the I^2C module, in addition to setting the **GPIOAFSEL** register bits for the I^2C clock and data pins, the data pins should be set to open drain using the **GPIO Open Drain Select (GPIOODR)** register (see examples in "Initialization and Configuration" on page 753).

GPIO Alternate Function Select (GPIOAFSEL)

GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (AHB) base: 0x4006.0000
GPIO Port K (AHB) base: 0x4006.1000
GPIO Port L (AHB) base: 0x4006.2000
GPIO Port M (AHB) base: 0x4006.3000
GPIO Port N (AHB) base: 0x4006.4000
GPIO Port P (AHB) base: 0x4006.5000
GPIO Port Q (AHB) base: 0x4006.6000
Offset 0x420
Type RW, reset -



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	AFSEL	RW	-	GPIO Alternate Function Select

Value	Description
0	The associated pin functions as a GPIO and is controlled by the GPIO registers.
1	The associated pin functions as a peripheral signal and is controlled by the alternate hardware function.

The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in Table 10-1 on page 743.

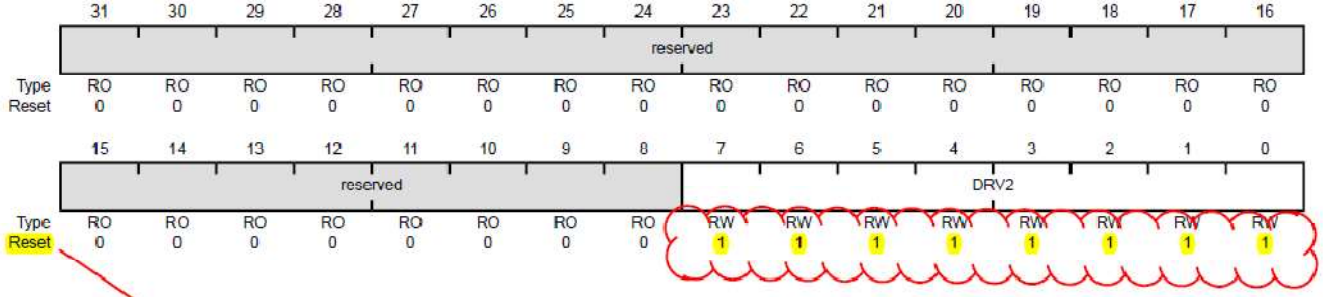
Register 11: GPIO 2-mA Drive Select (GPIODR2R), offset 0x500

The **GPIODR2R** register is the 2-mA drive control register. Each GPIO signal in the port can be individually configured without affecting the other pads. When setting the DRV2 bit for a GPIO signal, the corresponding DRV4 bit in the **GPIODR4R** register and DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware. By default, all GPIO pins have 2-mA drive.

Note: This register has no effect on port pins PL6 and PL7.

GPIO 2-mA Drive Select (GPIODR2R)

GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (AHB) base: 0x4006.0000
GPIO Port K (AHB) base: 0x4006.1000
GPIO Port L (AHB) base: 0x4006.2000
GPIO Port M (AHB) base: 0x4006.3000
GPIO Port N (AHB) base: 0x4006.4000
GPIO Port P (AHB) base: 0x4006.5000
GPIO Port Q (AHB) base: 0x4006.6000
Offset 0x500
Type RW, reset 0x0000.00FF



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV2	RW	0xFF	Output Pad 2-mA Drive Enable

Value	Description
0	The drive for the corresponding GPIO pin is controlled by the **GPIODR4R** or **GPIODR8R** register.
1	The corresponding GPIO pin has 2-mA drive.

Setting a bit in either the **GPIODR4** register or the **GPIODR8** register clears the corresponding 2-mA enable bit. The change is effective on the next clock cycle.

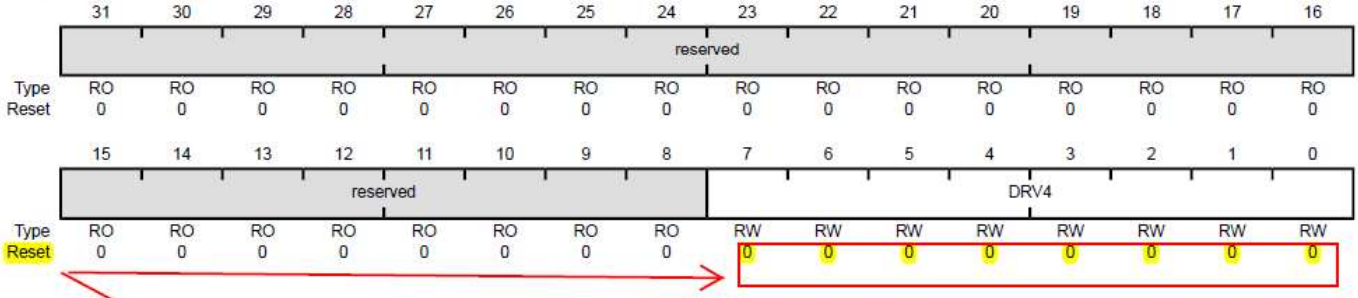
Register 12: GPIO 4-mA Drive Select (GPIODR4R), offset 0x504

The **GPIODR4R** register is the 4-mA drive control register. Each GPIO signal in the port can be individually configured without affecting the other pads. When setting the DRV4 bit for a GPIO signal, the corresponding DRV2 bit in the **GPIODR2R** register and DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware.

Note: This register has no effect on port pins PL6 and PL7.

GPIO 4-mA Drive Select (GPIODR4R)

GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (AHB) base: 0x4006.0000
GPIO Port K (AHB) base: 0x4006.1000
GPIO Port L (AHB) base: 0x4006.2000
GPIO Port M (AHB) base: 0x4006.3000
GPIO Port N (AHB) base: 0x4006.4000
GPIO Port P (AHB) base: 0x4006.5000
GPIO Port Q (AHB) base: 0x4006.6000
Offset 0x504
Type RW, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV4	RW	0x00	Output Pad 4-mA Drive Enable

Value	Description
0	The drive for the corresponding GPIO pin is controlled by the **GPIODR2R** or **GPIODR8R** register.
1	The corresponding GPIO pin has 4-mA drive.

Setting a bit in either the **GPIODR2** register or the **GPIODR8** register clears the corresponding 4-mA enable bit. The change is effective on the next clock cycle.

26.4 GPIO Pins and Alternate Functions

Table 26-5. GPIO Pins and Alternate Functions

IO	Pin	Analog or Special Function[a]	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[b]											
			1	2	3	4	5	6	7	8	11	13	14	15
PA0	33	-	U0Rx	I2C9SCL	T0CCP0	-	-	-	CAN0Rx	-	-	-	-	-
PA1	34	-	U0Tx	I2C9SDA	T0CCP1	-	-	-	CAN0Tx	-	-	-	-	-
PA2	35	-	U4Rx	I2C8SCL	T1CCP0	-	-	-	-	-	-	-	-	SSI0Clk
PA3	36	-	U4Tx	I2C8SDA	T1CCP1	-	-	-	-	-	-	-	-	SSI0Fss
PA4	37	-	U3Rx	I2C7SCL	T2CCP0	-	-	-	-	-	-	-	-	SSI0XDAT0
PA5	38	-	U3Tx	I2C7SDA	T2CCP1	-	-	-	-	-	-	-	-	SSI0XDAT1
PA6	40	-	U2Rx	I2C6SCL	T3CCP0	-	USB0EPEN	-	-	-	-	SSI0XDAT2	-	EPI0S8
PA7	41	-	U2Tx	I2C6SDA	T3CCP1	-	USB0PFLT	-	-	-	USB0EPEN	SSI0XDAT3	-	EPI0S9
PB0	95	USB0ID	U1Rx	I2C5SCL	T4CCP0	-	-	-	CAN1Rx	-	-	-	-	-
PB1	96	USB0VBUS	U1Tx	I2C5SDA	T4CCP1	-	-	-	CAN1Tx	-	-	-	-	-
PB2	91	-	-	I2C0SCL	T5CCP0	-	-	-	-	-	-	-	USB0STP	EPI0S27
PB3	92	-	-	I2C0SDA	T5CCP1	-	-	-	-	-	-	-	USB0CLK	EPI0S28
PB4	121	AIN10	U0CTS	I2C5SCL	-	-	-	-	-	-	-	-	-	SSI1Fss
PB5	120	AIN11	U0RTS	I2C5SDA	-	-	-	-	-	-	-	-	-	SSI1Clk
PC0	100	-	TCK SWCLK	-	-	-	-	-	-	-	-	-	-	-
PC1	99	-	TMS SWDIO	-	-	-	-	-	-	-	-	-	-	-
PC2	98	-	TDI	-	-	-	-	-	-	-	-	-	-	-
PC3	97	-	TDO SWO	-	-	-	-	-	-	-	-	-	-	-
PC4	25	C1-	U7Rx	-	-	-	-	-	-	-	-	-	-	EPI0S7
PC5	24	C1+	U7Tx	-	-	-	-	-	RTCCLK	-	-	-	-	EPI0S6
PC6	23	C0+	U5Rx	-	-	-	-	-	-	-	-	-	-	EPI0S5
PC7	22	C0-	U5Tx	-	-	-	-	-	-	-	-	-	-	EPI0S4
PD0	1	AIN15	-	I2C7SCL	T0CCP0	-	C0o	-	-	-	-	-	-	SSI2XDAT1
PD1	2	AIN14	-	I2C7SDA	T0CCP1	-	C1o	-	-	-	-	-	-	SSI2XDAT0
PD2	3	AIN13	-	I2C8SCL	T1CCP0	-	C2o	-	-	-	-	-	-	SSI2Fss
PD3	4	AIN12	-	I2C8SDA	T1CCP1	-	-	-	-	-	-	-	-	SSI2Clk
PD4	125	AIN7	U2Rx	-	T3CCP0	-	-	-	-	-	-	-	-	SSI1XDAT2
PD5	126	AIN6	U2Tx	-	T3CCP1	-	-	-	-	-	-	-	-	SSI1XDAT3
PD6	127	AIN5	U2RTS	-	T4CCP0	-	USB0EPEN	-	-	-	-	-	-	SSI2XDAT3
PD7	128	AIN4	U2CTS	-	T4CCP1	-	USB0PFLT	-	-	NMI	-	-	-	SSI2XDAT2
PE0	15	AIN3	U1RTS	-	-	-	-	-	-	-	-	-	-	-
PE1	14	AIN2	U1DSR	-	-	-	-	-	-	-	-	-	-	-
PE2	13	AIN1	U1DCD	-	-	-	-	-	-	-	-	-	-	-
PE3	12	AIN0	U1DTR	-	-	-	-	-	-	-	-	-	-	-
PE4	123	AIN9	U1RI	-	-	-	-	-	-	-	-	-	-	SSI1XDAT0

IO	Pin	Analog or Special Function[a]	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[b]											
			1	2	3	4	5	6	7	8	11	13	14	15
PE5	124	AIN8	-	-	-	-	-	-	-	-	-	-	-	SSI1XDAT1
PF0	42	-	-	-	-	-	EN0LED0	M0PWM0	-	-	-	-	SSI3XDAT1	TRD2
PF1	43	-	-	-	-	-	EN0LED2	M0PWM1	-	-	-	-	SSI3XDAT0	TRD1
PF2	44	-	-	-	-	-	-	M0PWM2	-	-	-	-	SSI3Fss	TRD0
PF3	45	-	-	-	-	-	-	M0PWM3	-	-	-	-	SSI3Clk	TRCLK
PF4	46	-	-	-	-	-	EN0LED1	M0FAULT0	-	-	-	-	SSI3XDAT2	TRD3
PG0	49	-	-	I2C1SCL	-	-	EN0PPS	M0PWM4	-	-	-	-	-	EPI0S11
PG1	50	-	-	I2C1SDA	-	-	-	M0PWM5	-	-	-	-	-	EPI0S10
PH0	29	-	U0RTS	-	-	-	-	-	-	-	-	-	-	EPI0S0
PH1	30	-	U0CTS	-	-	-	-	-	-	-	-	-	-	EPI0S1
PH2	31	-	U0DCD	-	-	-	-	-	-	-	-	-	-	EPI0S2
PH3	32	-	U0DSR	-	-	-	-	-	-	-	-	-	-	EPI0S3
PJ0	116	-	U3Rx	-	-	-	EN0PPS	-	-	-	-	-	-	-
PJ1	117	-	U3Tx	-	-	-	-	-	-	-	-	-	-	-
PK0	18	AIN16	U4Rx	-	-	-	-	-	-	-	-	-	-	EPI0S0
PK1	19	AIN17	U4Tx	-	-	-	-	-	-	-	-	-	-	EPI0S1
PK2	20	AIN18	U4RTS	-	-	-	-	-	-	-	-	-	-	EPI0S2
PK3	21	AIN19	U4CTS	-	-	-	-	-	-	-	-	-	-	EPI0S3
PK4	63	-	-	I2C3SCL	-	-	EN0LED0	M0PWM6	-	-	-	-	-	EPI0S32
PK5	62	-	-	I2C3SDA	-	-	EN0LED2	M0PWM7	-	-	-	-	-	EPI0S31
PK6	61	-	-	I2C4SCL	-	-	EN0LED1	M0FAULT1	-	-	-	-	-	EPI0S25
PK7	60	-	U0RI	I2C4SDA	-	-	RTCCLK	M0FAULT2	-	-	-	-	-	EPI0S24
PL0	81	-	-	I2C2SDA	-	-	-	M0FAULT3	-	-	-	-	USB0D0	EPI0S16
PL1	82	-	-	I2C2SCL	-	-	-	PhA0	-	-	-	-	USB0D1	EPI0S17
PL2	83	-	-	-	-	-	C0o	PhB0	-	-	-	-	USB0D2	EPI0S18
PL3	84	-	-	-	-	-	C1o	IDX0	-	-	-	-	USB0D3	EPI0S19
PL4	85	-	-	-	T0CCP0	-	-	-	-	-	-	-	USB0D4	EPI0S26
PL5	86	-	-	-	T0CCP1	-	-	-	-	-	-	-	USB0D5	EPI0S33
PL6	94	USB0DP	-	-	T1CCP0	-	-	-	-	-	-	-	-	-
PL7	93	USB0DM	-	-	T1CCP1	-	-	-	-	-	-	-	-	-
PM0	78	-	-	-	T2CCP0	-	-	-	-	-	-	-	-	EPI0S15
PM1	77	-	-	-	T2CCP1	-	-	-	-	-	-	-	-	EPI0S14
PM2	76	-	-	-	T3CCP0	-	-	-	-	-	-	-	-	EPI0S13
PM3	75	-	-	-	T3CCP1	-	-	-	-	-	-	-	-	EPI0S12
PM4	74	TMPR3	U0CTS	-	T4CCP0	-	-	-	-	-	-	-	-	-
PM5	73	TMPR2	U0DCD	-	T4CCP1	-	-	-	-	-	-	-	-	-
PM6	72	TMPR1	U0DSR	-	T5CCP0	-	-	-	-	-	-	-	-	-
PM7	71	TMPR0	U0RI	-	T5CCP1	-	-	-	-	-	-	-	-	-
PN0	107	-	U1RTS	-	-	-	-	-	-	-	-	-	-	-

A.S.H.D.

IO	Pin	Analog or Special Function[a]	Digital Function (GPIOPCTL PMCx Bit Field Encoding)[b]											
			1	2	3	4	5	6	7	8	11	13	14	15
PN1	108	-	U1CTS	-	-	-	-	-	-	-	-	-	-	-
PN2	109	-	U1DCD	U2RTS	-	-	-	-	-	-	-	-	-	EPI0S29
PN3	110	-	U1DSR	U2CTS	-	-	-	-	-	-	-	-	-	EPI0S30
PN4	111	-	U1DTR	U3RTS	I2C2SDA	-	-	-	-	-	-	-	-	EPI0S34
PN5	112	-	U1RI	U3CTS	I2C2SCL	-	-	-	-	-	-	-	-	EPI0S35
PP0	110	C2+	U6Rx	-	-	-	-	-	-	-	-	-	-	SSIXDAT2
PP1	119	C2-	U6Tx											SSIXDAT3
PP2	103		U0DTR										USB0NXT	EPI0S29
PP3	104		U1CTS	U0DCD					RTCCLK				USB0DIR	EPI0S30
PP4	105		U3RTS	U0DSR									USB0D7	
PP5	106		U3CTS	I2C2SCL									USB0D6	
PQ0	5	-	-	-	-	-	-	-	-	-	-	-	SSI3Clk	EPI0S20
PQ1	6	-	-	-	-	-	-	-	-	-	-	-	SSI3Fss	EPI0S21
PQ2	11	-	-	-	-	-	-	-	-	-	-	-	SSI3XDAT0	EPI0S22
PQ3	27	-	-	-	-	-	-	-	-	-	-	-	SSI3XDAT1	EPI0S23
PQ4	102	-	U1Rx	-	-	-	-	-	DIVSCLK	-	-	-	-	-

a. The TMPRn signals are digital signals enabled and configured by the Hibernation module. All other signals listed in this column are analog signals.

b. The digital signals that are shaded gray are the power-on default values for the corresponding GPIO pin. Encodings 9, 10, and 12 are not used on this device.

Las siguientes terminales están protegidas y la modificación de su configuración sólo se realiza desbloqueando la seguridad de los registros LOCK y COMMIT del puerto en cuestión. Se recomienda no emplearlos (especialmente los correspondientes a la interfaz JTAG) a menos que sea muy necesario y siguiendo todas las recomendaciones del fabricante.

Table 10-8. GPIO Pins With Special Considerations

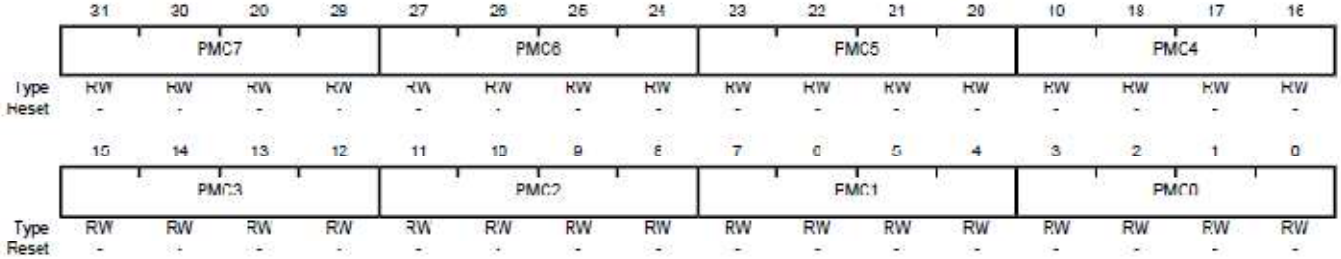
GPIO Pins	Default Reset State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL	GPIOCR
PC[3:0]	JTAG/SWD	1	1	0	1	0x1	0
PD[7]	GPIO[a]	0	0	0	0	0x0	0
PE[7]	GPIO[a]	0	0	0	0	0x0	0

a. This pin is configured as a GPIO by default but is locked and can only be reprogrammed by unlocking the pin in the GPIOLOCK register and uncommitting it by setting the GPIOCR register.

Register 22: GPIO Port Control (GPIOPCTL), offset 0x52C

The **GPIOPCTL** register is used in conjunction with the **GPIOAFSEL** register and selects the specific peripheral signal for each GPIO pin when using the alternate function mode. Most bits in the **GPIOAFSEL** register are cleared on reset, therefore most GPIO pins are configured as GPIOs by default. When a bit is set in the **GPIOAFSEL** register, the corresponding GPIO signal is controlled by an associated peripheral. The **GPIOPCTL** register selects one out of a set of peripheral functions for each GPIO, providing additional flexibility in signal definition. For information on the defined encodings for the bit fields in this register, refer to Table 26-5 on page 1817. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in the table below.

Note: If a particular input signal to a peripheral is assigned to two different GPIO port pins, the signal is assigned to the port with the lowest letter and the assignment to the higher letter port is ignored. If a particular output signal from a peripheral is assigned to two different GPIO port pins, the signal will output to both pins. Assigning an output signal from a peripheral to two different GPIO pins is not recommended.



Bit/Field	Name	Type	Reset	Description
31:28	PMC7	RW	-	Port Mux Control 7 This field controls the configuration for GPIO pin 7.
27:24	PMC6	RW	-	Port Mux Control 6 This field controls the configuration for GPIO pin 6.
23:20	PMC5	RW	-	Port Mux Control 5 This field controls the configuration for GPIO pin 5.
19:16	PMC4	RW	-	Port Mux Control 4 This field controls the configuration for GPIO pin 4.
15:12	PMC3	RW	-	Port Mux Control 3 This field controls the configuration for GPIO pin 3.
11:8	PMC2	RW	-	Port Mux Control 2 This field controls the configuration for GPIO pin 2.
7:4	PMC1	RW	-	Port Mux Control 1 This field controls the configuration for GPIO pin 1.
3:0	PMC0	RW	-	Port Mux Control 0 This field controls the configuration for GPIO pin 0.