Micro Processor & Controller

TMS320F28335 – Concerto
General Purpose I/O

Features

General-Purpose Input/Output (GPIO)

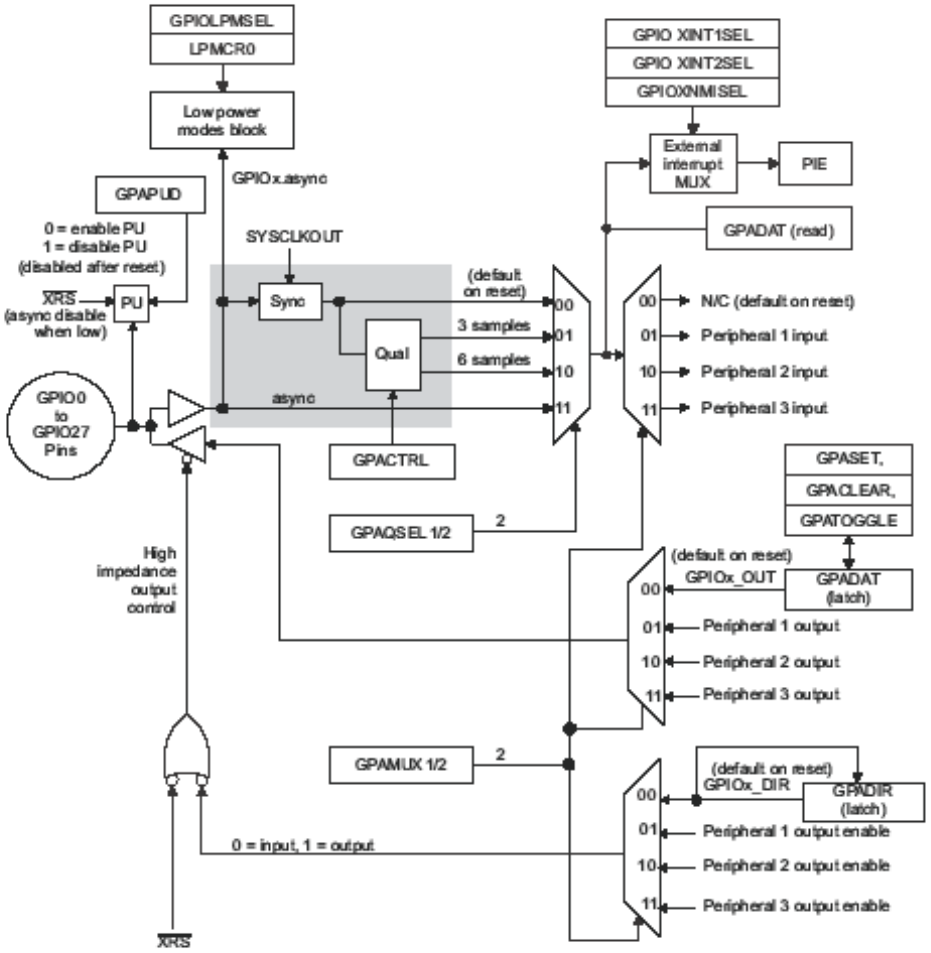
The GPIO multiplexing (MUX) registers are used to select the operation of shared pins. The pins are named by their general purpose I/O name (i.e., GPIO0 - GPIO87). These pins can be individually selected to operate as digital I/O, referred to as GPIO, or connected to one of up to three peripheral I/O signals (via the GPxMUXn registers). If selected for digital I/O mode, registers are provided to configure the pin direction (via the GPxDIR registers). You can also qualify the input signals to remove unwanted noise (via the GPxQSELn, GPACTRL, and GPBCTRL registers).

Module Overview (GPIO0-GPIO27)

4.1 GPIO Module Overview

Up to three independent peripheral signals are multiplexed on a single GPIO-enabled pin in addition to individual pin bit I/O capability. There are three 32-bit I/O ports. Port A consists of GPIO0-GPIO31, port B consists of GPIO32-GPIO63, and port C consists of GPIO64-87. Figure 4-1 shows the basic modes of operation for the GPIO module.

Figure 4-1. GPIO0 to GPIO27 Multiplexing Diagram



A GPxDAT latch/read are accessed at the same memory location.

GPIO Control Register – Port A, B, C

Table 4-1. GPIO Control Registers

	Name [1]	Address	Size (x16)	Register Description
A	GPACTRL	0x6F80	2	GPIO A Control Register (GPIO0-GPIO31)
	GPAQSEL1	0x6F82	2	GPIO A Qualifier Select 1 Register (GPIO0-GPIO15)
	GPAQSEL2	0x6F84	2	GPIO A Qualifier Select 2 Register (GPIO16-GPIO31)
	GPAMUX1	0x6F86	2	GPIO A MUX 1 Register (GPIO0-GPIO15)
	GPAMUX2	0x6F88	2	GPIO A MUX 2 Register (GPIO16-GPIO31)
	GPADIR	0x6F8A	2	GPIO A Direction Register (GPIO0-GPIO31)
	GPAPUD	0x6F8C	2	GPIO A Pull Up Disable Register (GPIO0-GPIO31)
B	GPBCTRL	0x6F90	2	GPIO B Control Register (GPIO32-GPIO63)
	GPBQSEL1	0x6F92	2	GPIO B Qualifier Select 1 Register (GPIO32-GPIO47)
	GPBQSEL2	0x6F94	2	GPIO B Qualifier Select 2 Register (GPIO48 - GPIO63)
	GPBMUX1	0x6F96	2	GPIO B MUX 1 Register (GPIO32-GPIO47)
	GPBMUX2	0x6F98	2	GPIO B MUX 2 Register (GPIO48-GPIO63)
	GPBDIR	0x6F9A	2	GPIO B Direction Register (GPIO32-GPIO63)
	GPBPUD	0x6F9C	2	GPIO B Pull Up Disable Register (GPIO32-GPIO63)
C	GPCMUX1	0x6FA6	2	GPIO C MUX 1 Register (GPIO64-GPIO79)
	GPCMUX2	0x6FA8	2	GPIO C MUX 2 Register (GPIO80-GPIO87)
	GPCDIR	0x6FAA	2	GPIO C Direction Register (GPIO64-GPIO87)
	GPCPUD	0x6FAC	2	GPIO C Pull Up Disable Register (GPIO64-GPIO87)

GPIO INT Register (GPIO0-GPIO63)

Table 4-2. GPIO Interrupt and Low Power Mode Select Registers

Name [(1)]	Address	Size (x16)	Register Description
GPIOXINT1SEL	0x6FE0	1	XINT1 Source Select Register (GPIO0-GPIO31)
GPIOXINT2SEL	0x6FE1	1	XINT2 Source Select Register (GPIO0-GPIO31)
GPIOXNMISEL	0x6FE2	1	XNMI Source Select Register (GPIO0-GPIO31)
GPIOXINT3SEL	0x6FE3	1	XINT3 Source Select Register (GPIO32 - GPIO63)
GPIOXINT4SEL	0x6FE4	1	XINT4 Source Select Register (GPIO32 - GPIO63)
GPIOXINT5SEL	0x6FE5	1	XINT5 Source Select Register (GPIO32 - GPIO63)
GPIOXINT6SEL	0x6FE6	1	XINT6 Source Select Register (GPIO32 - GPIO63)
GPIOXINT7SEL	0x6FE7	1	XINT7 Source Select Register (GPIO32 - GPIO63)
GPIOLPMSEL	0x6FE8	1	LPM wakeup Source Select Register (GPIO0-GPIO31)

GPIO Data Register – Port A, B, C

Table 4-3. GPIO Data Registers

Name	Address	Size (x16)	Register Description
GPADAT	0x6FC0	2	GPIO A Data Register (GPIO0-GPIO31)
GPASET	0x6FC2	2	GPIO A Set Register (GPIO0-GPIO31)
GPACLEAR	0x6FC4	2	GPIO A Clear Register (GPIO0-GPIO31)
GPATOGGLE	0x6FC6	2	GPIO A Toggle Register (GPIO0-GPIO31)
GPBDAT	0x6FC8	2	GPIO B Data Register (GPIO32-GPIO63)
GPBSET	0x6FCA	2	GPIO B Set Register (GPIO32-GPIO63)
GPBCLEAR	0x6FCC	2	GPIO B Clear Register (GPIO32-GPIO63)
GPBTOGGLE	0x6FCE	2	GPIO B Toggle Register (GPIO32-GPIO63)
GPCDAT	0x6FD0	2	GPIO C Data Register (GPIO64 - GPIO87)
GPCSET	0x6FD2	2	GPIO C Set Register (GPIO64 - GPIO87)
GPCCLEAR	0x6FD4	2	GPIO C Clear Register (GPIO64 - GPIO87)
GPCTOGGLE	0x6FD6	2	GPIO C Toggle Register (GPIO64 - GPIO87)

C – Init GPIO Pins

```
#include "DSP2833x_Device.h"     // DSP2833x Headerfile Include File
#include "DSP2833x_Examples.h"   // DSP2833x Examples Include File

//---------------------------------------------------------------------------
// InitGpio:
//---------------------------------------------------------------------------
// This function initializes the Gpio to a known (default) state.
//
// For more details on configuring GPIO's as peripheral functions,
// refer to the individual peripheral examples and/or GPIO setup example.
void InitGpio(void)
{
   EALLOW;

   // Each GPIO pin can be:
   // a) a GPIO input/output
   // b) peripheral function 1
   // c) peripheral function 2
   // d) peripheral function 3
   // By default, all are GPIO Inputs
   GpioCtrlRegs.GPAMUX1.all = 0x0000;     // GPIO functionality GPIO0-GPIO15
   GpioCtrlRegs.GPAMUX2.all = 0x0000;     // GPIO functionality GPIO16-GPIO31
   GpioCtrlRegs.GPBMUX1.all = 0x0000;     // GPIO functionality GPIO32-GPIO39
   GpioCtrlRegs.GPBMUX2.all = 0x0000;     // GPIO functionality GPIO48-GPIO63
   GpioCtrlRegs.GPCMUX1.all = 0x0000;     // GPIO functionality GPIO64-GPIO79
   GpioCtrlRegs.GPCMUX2.all = 0x0000;     // GPIO functionality GPIO80-GPIO95

   GpioCtrlRegs.GPADIR.all = 0x0000;      // GPIO0-GPIO31 are inputs
   GpioCtrlRegs.GPBDIR.all = 0x0000;      // GPIO32-GPIO63 are inputs
   GpioCtrlRegs.GPCDIR.all = 0x0000;      // GPI064-GPIO95 are inputs

   // Each input can have different qualification
   // a) input synchronized to SYSCLKOUT
   // b) input qualified by a sampling window
   // c) input sent asynchronously (valid for peripheral inputs only)
   GpioCtrlRegs.GPAQSEL1.all = 0x0000;    // GPIO0-GPIO15 Synch to SYSCLKOUT
   GpioCtrlRegs.GPAQSEL2.all = 0x0000;    // GPIO16-GPIO31 Synch to SYSCLKOUT
   GpioCtrlRegs.GPBQSEL1.all = 0x0000;    // GPIO32-GPIO39 Synch to SYSCLKOUT
   GpioCtrlRegs.GPBQSEL2.all = 0x0000;    // GPIO48-GPIO63 Synch to SYSCLKOUT

   // Pull-ups can be enabled or disabled.
   GpioCtrlRegs.GPAPUD.all = 0x0000;      // Pullup's enabled GPIO0-GPIO31
   GpioCtrlRegs.GPBPUD.all = 0x0000;      // Pullup's enabled GPIO32-GPIO63
   GpioCtrlRegs.GPCPUD.all = 0x0000;      // Pullup's enabled GPIO64-GPIO79

   //GpioCtrlRegs.GPAPUD.all = 0xFFFF;    // Pullup's disabled GPIO0-GPIO31

   //GpioCtrlRegs.GPBPUD.all = 0xFFFF;    // Pullup's disabled GPIO32-GPIO34
   //GpioCtrlRegs.GPCPUD.all = 0xFFFF     // Pullup's disabled GPIO64-GPIO79

   EDIS;
}
```

C – Using GPIO Data Registers

Attention
Hazard situation
using
DAT registers

```
//###########################################################################
//
// FILE:    Flax_DelfinoEvbGpioToggle.c
//
// TITLE:   DSP2833x Device GPIO toggle test program.
// This template was written by Eli Flaxer for DelfinoEvb Evaluation Board
//
//
//###########################################################################
// $TI Release: 2833x/2823x Header Files and Peripheral Examples V133 $
// $Release Date: June 8, 2012 $
//###########################################################################

#include "DSP28x_Project.h"     // Device Header file and Examples Include File
#include "LCD2x16Display.h"

void DelfinoEvbGpioSelect(void);
void MyMainProg(void);

int32 MyDelayLoop = 200000L;

/*****************************************************************************/
void GpioCSetClear(int k,int x)
{
    if (x)
        GpioDataRegs.GPCSET.all = (1L<<k);
    else
        GpioDataRegs.GPCCLEAR.all = (1L<<k);
}
/*****************************************************************************/
void main(void)
{

// Step 1. Initialize System Control:
// PLL, WatchDog, enable Peripheral Clocks
// This example function is found in the DSP2833x_SysCtrl.c file.
   InitSysCtrl();

// Step 2. Initalize GPIO:
// This example function is found in the DSP2833x_Gpio.c file and
// illustrates how to set the GPIO to it's default state.
// InitGpio();  // Skipped for this example

// For this example use the following configuration:
   DelfinoEvbGpioSelect();

// Step 3. Clear all interrupts and initialize PIE vector table:
// Disable CPU interrupts
   DINT;

// Initialize PIE control registers to their default state.
// The default state is all PIE interrupts disabled and flags
// are cleared.
// This function is found in the DSP2833x_PieCtrl.c file.
   InitPieCtrl();

// Disable CPU interrupts and clear all CPU interrupt flags:
```