MOD54415 PinIO Class

Application Note

Revision 1.0 | 2014-05-05 | Initial Release

Table of Contents

Introduction

The PinIO class provides an easy way to configure and operate the Freescale MCF54415 microprocessor GPIO signals. Each signal pin on the MCF54415 can have multiple functions. You can use the PinIO class to control GPIO signals without having to explicitly configure the MCF54415 registers. Configuration of the processor registers are done in the member functions of the PinIO class. Depending on the hardware revision and configuration, there can be up to 42 pins on the MOD54415 that are made available for GPIO. This document will list the pins that can be used for GPIO and how to use them.

If you do wish to access these registers directly, and assuming you have the NNDK tools installed, we recommend you use the register structure defined in:

```
\nburn\MOD5441X\include\sim5441x.h
```

and use the “MCF5441x Reference Manual” located in:

```
\nburn\docs\FreescaleManuals\MCF54415RM.pdf
```

to learn the operation of each register.

MOD54415 Revisions and Pin Functionality Changes

The functionality of some signal pins has changed between different hardware revisions. The changes are explained in the table below. The table lists the module hardware revision, what signal pins are affected, and the functionality assigned to them.

Hardware Revision	Signal Pin	Primary Function Assignment
1.5 (Initial Release) 1.6	J1-9	Byte Enable 2 (CPU pin E2)
	J1-10	Byte Enable 3 (CPU pin C1)
	J2-17	SSI 0 – Serial Receive Data (CPU pin C12)
	J2-18	SSI 0 – Serial Transmit Data (CPU pin C13)
1.7	J1-9	(No change from v1.6)
	J1-10	(No change from v1.6)
	J2-17	USB- On-the-Go [CPU pin A14; component R36 stuffed (default)]* USB- Host (CPU pin A15; component R34 stuffed)*
	J2-18	USB+ On-the-Go [CPU pin B14; component R35 stuffed (default)]* USB+ Host (CPU pin B15; component R33 stuffed)*
1.8	J1-9	Byte Enable 1 (CPU pin D1)
	J1-10	Byte Enable 0 (CPU pin F4)
	J2-17	(No change from v1.7)
	J2-18	(No change from v1.7)
1.9	J1-9	(No change from v1.8)
	J1-10	(No change from v1.8)
	J2-17	USB- On-the-Go [CPU pin A14; component R36 stuffed (default)]* USB- Host (CPU pin A15; component R38 stuffed)* SSI 0 – Serial Receive Data (CPU pin C12; component R81 stuffed)
	J2-18	USB+ On-the-Go [CPU pin B14; component R35 stuffed (default)]* USB+ Host (CPU pin B15; component R37 stuffed)* SSI 0 – Serial Transmit Data (CPU pin C13; component R80 stuffed)

* Not configurable for GPIO functionality. Which primary function is associated with the signal pin depends on how the jumper resistors are stuffed on the module.

Electrical Specifications

The current drive capabilities of the GPIO pins are the same for all pins. The instantaneous maximum current for a single pin is 25 mA. The sustained current drive is 5 mA. Please see the document "MCF5441x ColdFire Microprocessor Data Sheet" located in:

```
\nburn\docs\FreescaleManuals\MCF54415DataSheet.pdf
```

for more information.

PinIO Class

This class is defined in the header file `\nburn\include\pins.h`. With this class, the pins can be configured for their primary function, alternate function(s), or GPIO. If the pins are set for GPIO, then you can set, clear, read the state of the pins, drive the pins, or set them for high impedance by simply using the appropriate member function. The supported functions defined for each pin and the member functions to use those pins (when configured for GPIO) are respectively listed in the following files:

```
\nburn\MOD5441X\include\pinconstant.h
\nburn\MOD5441X\system\pins.cpp
```

PinIO Class Constants

The table below lists the 42 pins available for GPIO on the MOD54415 (hardware revisions 1.7 and 1.8 are restricted to 40 pins available), as well as their primary and alternate functions, if any. Note that some functions may not be supported on certain module hardware revisions; what revisions are supported is indicated in the "Connector" column for a particular pin.

Connector	Pin	Definition	Function
J1	5	PINJ1_5_FB_CS1	3: General Purpose Chip Select 1
		PINJ1_5_NFC_CE	1: NAND Flash Controller – Chip Enable
		PINJ1_5_GPIO	0: Port B-4 GPIO
J1	6	PINJ1_6_FB_CS4	3: General Purpose Chip Select 4
		PINJ1_6_DREQ1	2: External DMA Request 1
		PINJ1_6_GPIO	0: Port B-5 GPIO
J1	7	PINJ1_7_FB_CS5	3: General Purpose Chip Select 5
		PINJ1_7_DACK1	2: External DMA Acknowledge 1
		PINJ1_7_GPIO	0: Port B-6 GPIO
J1 (v1.5-1.7)	9	PINJ1_9_FB_BE2	3: Byte Enable 2
		PINJ1_9_FB_CS2	2: General Purpose Chip Select 2
		PINJ1_9_FB_A0	1: Address 0 / NAND Flash Controller – Cmd Latch En
		PINJ1_9_GPIO	0: Port A-2 GPIO
J1 (v1.5-1.7)	10	PINJ1_10_FB_BE3	3: Byte Enable 3
		PINJ1_10_FB_CS3	2: General Purpose Chip Select 3
		PINJ1_10_FB_A1	1: Address 1 / NAND Flash Controller – Addr Latch En
		PINJ1_10_GPIO	0: Port A-3 GPIO
J1 (v1.8+)	9	PINJ1_9_FB_BE1	3: Byte Enable 1
		PINJ1_9_FB_TSIZ1	2: Transfer Size 1
		PINJ1_9_GPIO	0: Port A-1 GPIO
J1 (v1.8+)	10	PINJ1_10_FB_BE0	3: Byte Enable 0
		PINJ1_10_FB_TSIZ0	2: Transfer Size 0
		PINJ1_10_GPIO	0: Port A-0 GPIO
J1	13	PINJ1_13_FB_TA	3: Transfer Acknowledge
		PINJ1_13_NFC_RB	1: NAND Flash Controller – Flash Ready/Busy
		PINJ1_13_GPIO	0: Port A-4 GPIO

J1	31	PINJ1_31_FB_CLK	3: Internal Bus Clock
		PINJ1_31_GPIO	0: Port B-7 GPIO
J2	3	PINJ2_3_UART0_RXD	3: UART 0 – Receive
		PINJ2_3_I2C4_SDA	2: I2C 4 – Serial Data
		PINJ2_3_DSPI2_SIN	1: DSPI 2 – Serial Data In
		PINJ2_3_GPIO	0: Port F-4 GPIO
J2	4	PINJ2_4_UART0_TXD	3: UART 0 – Transmit
		PINJ2_4_I2C4_SCL	2: I2C 4 – Serial Clock
		PINJ2_4_DSPI2_SOUT	1: DSPI 2 – Serial Data Out
		PINJ2_4_GPIO	0: Port F-3 GPIO
J2	15	PINJ2_15_SSI0_MCLK	3: SSI 0 – Serial Master Clock
		PINJ2_15_SSI_CLKIN	2: SSI Clock Input
		PINJ2_15_SIM1_CLK	1: SIM 1 – Clock
		PINJ2_15_GPIO	0: Port H-4 GPIO
J2	16	PINJ2_16_SSI0_BCLK	3: SSI 0 – Serial Bit Clock
		PINJ2_16_UART7_RXD	2: UART 7 – Receive
		PINJ2_16_SIM1_PD	1: SIM 1 – Card Insertion Detect Signal
		PINJ2_16_GPIO	0: Port H-3 GPIO
J2 (v1.5-1.6, 1.9+)	17	PINJ2_17_SSI0_RXD	3: SSI 0 – Serial Receive Data
		PINJ2_17_I2C2_SDA	2: I2C 2 – Serial Data
		PINJ2_17_SIM1_VEN	1: SIM 1 – Power Supply Enable Signal
		PINJ2_17_GPIO	0: Port H-7 GPIO
J2 (v1.5-1.6, 1.9+)	18	PINJ2_18_SSI0_TXD	3: SSI 0 – Serial Transmit Data
		PINJ2_18_I2C2_SCL	2: I2C 2 – Serial Clock
		PINJ2_18_SIM1_DATA	1: SIM 1 – Bidirectional Transmit/Receive Data Signal
		PINJ2_18_GPIO	0: Port H-6 GPIO
J2	19	PINJ2_19_UART2_TXD	3: UART 2 – Transmit
		PINJ2_19_PWM_B3	2: PWM B3 – Output Signal/Input Capture
		PINJ2_19_SSI1_TXD	1: SSI 1 – Serial Transmit Data
		PINJ2_19_GPIO	0: Port E-3 GPIO
J2	20	PINJ2_20_SSI0_FS	3: SSI 0 – Serial Frame Sync
		PINJ2_20_UART7_TXD	2: UART 7 – Transmit
		PINJ2_20_SIM1_RST	1: SIM 1 – Reset Signal
		PINJ2_20_GPIO	0: Port H-5 GPIO
J2	21	PINJ2_21_UART1_RXD	3: UART 1 – Receive
		PINJ2_21_I2C5_SDA	2: I2C 5 – Serial Data
		PINJ2_21_DSPI3_SIN	1: DSPI 3 – Serial Data In
		PINJ2_21_GPIO	0: Port E-0 GPIO
J2	22	PINJ2_22_UART1_TXD	3: UART 1 – Transmit
		PINJ2_22_I2C5_SCL	2: I2C 5 – Serial Clock
		PINJ2_22_DSPI3_SOUT	1: DSPI 3 – Serial Data Out
		PINJ2_22_GPIO	0: Port F-7 GPIO
J2	23	PINJ2_23_UART1_RTS	3: UART 1 – Request to Send
		PINJ2_23_UART5_RXD	2: UART 5 – Receive
		PINJ2_23_DSPI3_PCS0	1: DSPI 3 – Peripheral Chip Select 0
		PINJ2_23_GPIO	0: Port E-1 GPIO / Rapid GPIO 8
J2	24	PINJ2_24_UART1_CTS	3: UART 1 – Clear to Send
		PINJ2_24_UART5_TXD	2: UART 5 – Transmit

		PINJ2_24_DSPI3_SCK	1: DSPI 3 – Serial Clock
		PINJ2_24_GPIO	0: Port E-2 GPIO / Rapid GPIO 7
J2	25	PINJ2_25_SDHC_CLK	3: SDHC Clock
		PINJ2_25_PWM_A0	2: PWM A0 – Output Signal/Input Capture
		PINJ2_25_DSPI1_SCK	1: DSPI 1 – Serial Clock
		PINJ2_25_GPIO	0: Port G-5 GPIO
J2	26	PINJ2_26_IRQ3	3: External Interrupt 3
		PINJ2_26_DSPI0_PCS3	2: DSPI 0 – Peripheral Chip Select 3
		PINJ2_26_USBH_VBUS_EN	1: USB Host VBUS Enable
		PINJ2_26_GPIO	0: Port C-3 GPIO
J2	27	PINJ2_27_SDHC_CMD	3: SDHC Command Line
		PINJ2_27_PWM_B0	2: PWM B0 – Output Signal/Input Capture
		PINJ2_27_DSPI1_SIN	1: DSPI 1 – Serial Data In
		PINJ2_27_GPIO	0: Port G-6 GPIO
J2	28	PINJ2_28_SDHC_DAT0	3: SDHC DAT0 Line / Busy-State Detect
		PINJ2_28_PWM_B2	2: PWM B2 – Output Signal/Input Capture
		PINJ2_28_DSPI1_SOUT	1: DSPI 1 – Serial Data Out
		PINJ2_28_GPIO	0: Port G-7 GPIO
J2	29	PINJ2_29_UART0_CTS	3: UART 0 – Clear to Send
		PINJ2_29_UART4_TXD	2: UART 4 – Transmit
		PINJ2_29_DSPI2_SCK	1: DSPI 2 – Serial Clock
		PINJ2_29_GPIO	0: Port F-6 GPIO / Rapid GPIO 5
J2	30	PINJ2_30_SDHC_DAT3	3: SDHC DAT3 Line / Card Detection
		PINJ2_30_PWM_A1	2: PWM A1 – Output Signal/Input Capture
		PINJ2_30_DSPI1_PCS0	1: DSPI 1 – Peripheral Chip Select 0
		PINJ2_30_GPIO	0: Port F-2 GPIO
J2	31	PINJ2_31_UART2_RXD	3: UART 2 – Receive
		PINJ2_31_PWM_A3	2: PWM A3 – Output Signal/Input Capture
		PINJ2_31_SSI1_RXD	1: SSI 1 – Serial Receive Data
		PINJ2_31_GPIO	0: Port E-4 GPIO
J2	32	PINJ2_32_T3IN	3: Timer Input 3
		PINJ2_32_T3OUT	2: Timer Output 3
		PINJ2_32_USBO_VBUS_EN	1: USB On-the-Go VBUS Enable
		PINJ2_32_GPIO	0: Port D-2 GPIO / Rapid GPIO 1
J2	33	PINJ2_33_T2IN	3: Timer Input 2
		PINJ2_33_T2OUT	2: Timer Output 2
		PINJ2_33_SDHC_DAT2	1: SDHC DAT2 Line / Read Wait
		PINJ2_33_GPIO	0: Port D-1 GPIO / Rapid GPIO 2
J2	34	PINJ2_34_T1IN	3: Timer Input 1
		PINJ2_34_T1OUT	2: Timer Output 1
		PINJ2_34_SDHC_DAT1	1: SDHC DAT1 Line / Interrupt Detect
		PINJ2_34_GPIO	0: Port D-0 GPIO / Rapid GPIO 3
J2	35	PINJ2_35_SDHC_DAT1	3: SDHC DAT1 Line / Interrupt Detect
		PINJ2_35_PWM_A2	2: PWM A2 – Output Signal/Input Capture
		PINJ2_35_DSPI1_PCS1	1: DSPI 1 – Peripheral Chip Select 1
		PINJ2_35_GPIO	0: Port F-0 GPIO
J2	36	PINJ2_36_T0IN	3: Timer Input 0
		PINJ2_36_T0OUT	2: Timer Output 0

		PINJ2_36_USBO_VBUS_OC	1: USB On-the-Go VBUS Over-Current
		PINJ2_36_GPIO	0: Port E-7 GPIO / Rapid GPIO 4
J2	37	PINJ2_37_OW_DAT	3: 1-Wire Data Signal
		PINJ2_37_DACK0	2: DMA Acknowledge 0
		PINJ2_37_GPIO	0: Port D-3 GPIO / Rapid GPIO 0
J2	38	PINJ2_38_UART0_RTS	3: UART 0 – Request to Send
		PINJ2_38_UART4_RXD	2: UART 4 – Receive
		PINJ2_38_DSPI2_PCS0	1: DSPI 2 – Peripheral Chip Select 0
		PINJ2_38_GPIO	0: Port F-5 GPIO / Rapid GPIO 6
J2	39	PINJ2_39_I2C0_SDA	3: I2C 0 – Serial Data
		PINJ2_39_UART8_RXD	2: UART 8 – Receive
		PINJ2_39_CAN0_RX	1: CAN 0 – Receive
		PINJ2_39_GPIO	0: Port B-1 GPIO
J2	40	PINJ2_40_SDHC_DAT2	3: SDHC DAT2 Line / Read Wait
		PINJ2_40_PWM_B1	2: PWM B1 – Output Signal/Input Capture
		PINJ2_40_DSPI1_PCS2	1: DSPI 1 – Peripheral Chip Select 2
		PINJ2_40_GPIO	0: Port F-1 GPIO
J2	41	PINJ2_41_CAN1_RX	3: CAN 1 – Receive
		PINJ2_41_UART9_RXD	2: UART 9 – Receive
		PINJ2_41_I2C1_SDA	1: I2C 1 – Serial Data
		PINJ2_41_GPIO	0: Port C-7 GPIO
J2	42	PINJ2_42_I2C0_SCL	3: I2C 0 – Serial Clock
		PINJ2_42_UART8_TXD	2: UART 8 – Transmit
		PINJ2_42_CAN0_TX	1: CAN 0 – Transmit
		PINJ2_42_GPIO	0: Port B-2 GPIO
J2	43	PINJ2_43_IRQ2	3: External Interrupt 2
		PINJ2_43_DSPI0_PCS2	2: DSPI 0 – Peripheral Chip Select 2
		PINJ2_43_USBH_VBUS_OC	1: USB Host VBUS Over-Current
		PINJ2_43_GPIO	0: Port C-2 GPIO
J2	44	PINJ2_44_CAN1_TX	3: CAN 1 – Transmit
		PINJ2_44_UART9_TXD	2: UART 9 – Transmit
		PINJ2_44_I2C1_SCL	1: I2C 1 – Serial Clock
		PINJ2_44_GPIO	0: Port B-0 GPIO
J2	45	PINJ2_45_IRQ1	3: External Interrupt 1
		PINJ2_45_GPIO	0: Port C-1 GPIO
J2	47	PINJ2_47_IRQ6	3: External Interrupt 6
		PINJ2_47_USB_CLKIN	1: USB Clock In
		PINJ2_47_GPIO	0: Port C-5 GPIO
J2	48	PINJ2_48_IRQ7	1: External Interrupt 7
		PINJ2_48_GPIO	0: Port C-6 GPIO

Pin Constants Table

The “Definition” column in the pin constants table above describes the values available for each pin when used with the PinIO class member function “function”. For example, if pin J2-30 was to be configured for GPIO, then it would be written as:

```
J2[30].function( PINJ2_30_GPIO );
```

Or, if I^2C 0 serial clock signal functionality is needed, then it would be written as:

```
J2[42].function( PINJ2_42_I2C0_SCL );
```

The “Function” column in the pin constants table describes the primary, alternate and GPIO functions for each pin. The numbers to the left represent the following (some pins that have only one alternate function may use either ‘1’ or ‘2’, while others that have no alternate function will use ‘1’ or ‘3’ for the primary function):

3: Primary Function
2: Alternate Function 1
1: Alternate Function 2
0: GPIO

These values are used to set the bits in a pin’s respective pin assignment register to configure for a specific function.

PinIO Class Member Functions

Using the PinIO class member functions to configure and use the GPIO pins eliminates the time and complexity of having to look up the proper documentation and use the right register and bits for a desired pin or set of pins. For example, to configure pin J2-44 (CAN 1 – Transmit) as GPIO and set it high without the PinIO class, it would be written like this:

```
#include <sim5441x.h>

sim1.gpio.par_cani2c &= ~0x0C;      // Configure pin J2-44 for GPIO
sim1.gpio.ppdsdr_b = 0x01;          // Set bit to be driven out on pin
sim1.gpio.pddr_b |= 0x01;           // Set signal direction as output
```

Knowing the right register and bits are not required with the PinIO class, thus making it more convenient:

```
#include <pins.h>

J2[44].function(PINJ2_44_GPIO);     // Configure pin J2-44 for GPIO
J2[44] = 1;                         // Drive pin as output high
```

The following lists the member functions that can be used with the PinIO class:

Member Function Name	Description	Example
`void set()`	Set output high	`J1[7].set();` `J1[7] = 1;`
`void clr()`	Set output low	`J2[21].clr();` `J2[21] = 0;`

`BOOL read()`	Read pin high/low state	`BOOL bpinstate = J2[30];`
		`if (!J2[30])`
		`  iprintf ("The pin is low");`

`void hiz()`	Set output to tristate (high impedance input)	`J2[38].hiz();`
`void drive`	Turn output on (opposite of tristate)	`J2[27].drive();`
`void function()`	Set pin to special function or GPIO	`J2[44].function(PINJ2_44_GPIO);` `J2[42].function(PINJ2_42_I2C0_SCL);`

Program Example

```
/*****************************************************************************
 * LED BINARY COUNTER
 *
 * This program configures multiple signal pins as GPIO outputs to utilize the
 * available general-purpose LEDs on the MOD-DEV-70CR development board as a
 * visual binary counter for the MOD54415 platform. The LED with the lowest
 * numerical designator ID is used as the least significant bit. The binary
 * counter increments once every second.  Once the counter reaches the maximum
 * value that the LEDs are capable of displaying, the counter will automatically
 * reset back to zero.
 */

#include "predef.h"
#include <stdio.h>
#include <ctype.h>
#include <startnet.h>
#include <autoupdate.h>
#include <dhcpclient.h>
#include <pins.h>

extern "C"
{
    void UserMain(void *pd);
}

const char *AppName = "MCF5441X-LedBinaryCounter";

/*
 * Translate the value of the counter into binary form to be represented on the
 * LEDs of the development board being used
 */
void incrementLeds(BYTE nCount)
{
    /*
     * On the MOD-DEV-70CR v1.93+, setting the pins turns off the LED, clearing
     * the pin turns on the LED
     */
    if (nCount & 0x01) J2[15] = 0;      // LED1
    else J2[15] = 1;

    if (nCount & 0x02) J2[16] = 0;      // LED2
    else J2[16] = 1;

    if (nCount & 0x04) J2[31] = 0;      // LED3
    else J2[31] = 1;

    if (nCount & 0x08) J2[23] = 0;      // LED4
    else J2[23] = 1;

    if (nCount & 0x10) J2[37] = 0;      // LED5
    else J2[37] = 1;
```

```
    if (nCount & 0x20) J2[19] = 0;      // LED6
    else J2[19] = 1;

    if (nCount & 0x40) J2[20] = 0;      // LED7
    else J2[20] = 1;

    if (nCount & 0x80) J2[24] = 0;      // LED8
    else J2[24] = 1;
}

/**
 * The main task
 */
void UserMain(void *pd)
{
    InitializeStack();
    if (EthernetIP == 0) GetDHCPAddress();
    OSChangePrio(MAIN_PRIO);
    EnableAutoUpdate();

    iprintf("Application started\r\n");

    /*
     * Configure MOD54415 pins J2-15, 16, 19, 20, 23, 24, 31, and 37 as GPIO.
     * Note that LED3 and LED5 in this example will work only with the
     * MOD-DEV-70CR v1.93 and later
     */
    J2[15].function(PINJ2_15_GPIO);     // MOD-DEV-70CR - LED1
    J2[16].function(PINJ2_16_GPIO);     // MOD-DEV-70CR - LED2
    J2[31].function(PINJ2_31_GPIO);     // MOD-DEV-70CR - LED3 (v1.93+)
    J2[23].function(PINJ2_23_GPIO);     // MOD-DEV-70CR - LED4
    J2[37].function(PINJ2_37_GPIO);     // MOD-DEV-70CR - LED5 (v1.93+)
    J2[19].function(PINJ2_19_GPIO);     // MOD-DEV-70CR - LED6
    J2[20].function(PINJ2_20_GPIO);     // MOD-DEV-70CR - LED7
    J2[24].function(PINJ2_24_GPIO);     // MOD-DEV-70CR - LED8

    BYTE nCounter = 0;      // For tracking the binary counting of the LEDs

    while (1) {
        /*
         * Increment the counter once every second
         */
        OSTimeDly(TICKS_PER_SECOND);
        incrementLeds(nCounter++);
    }
}
```