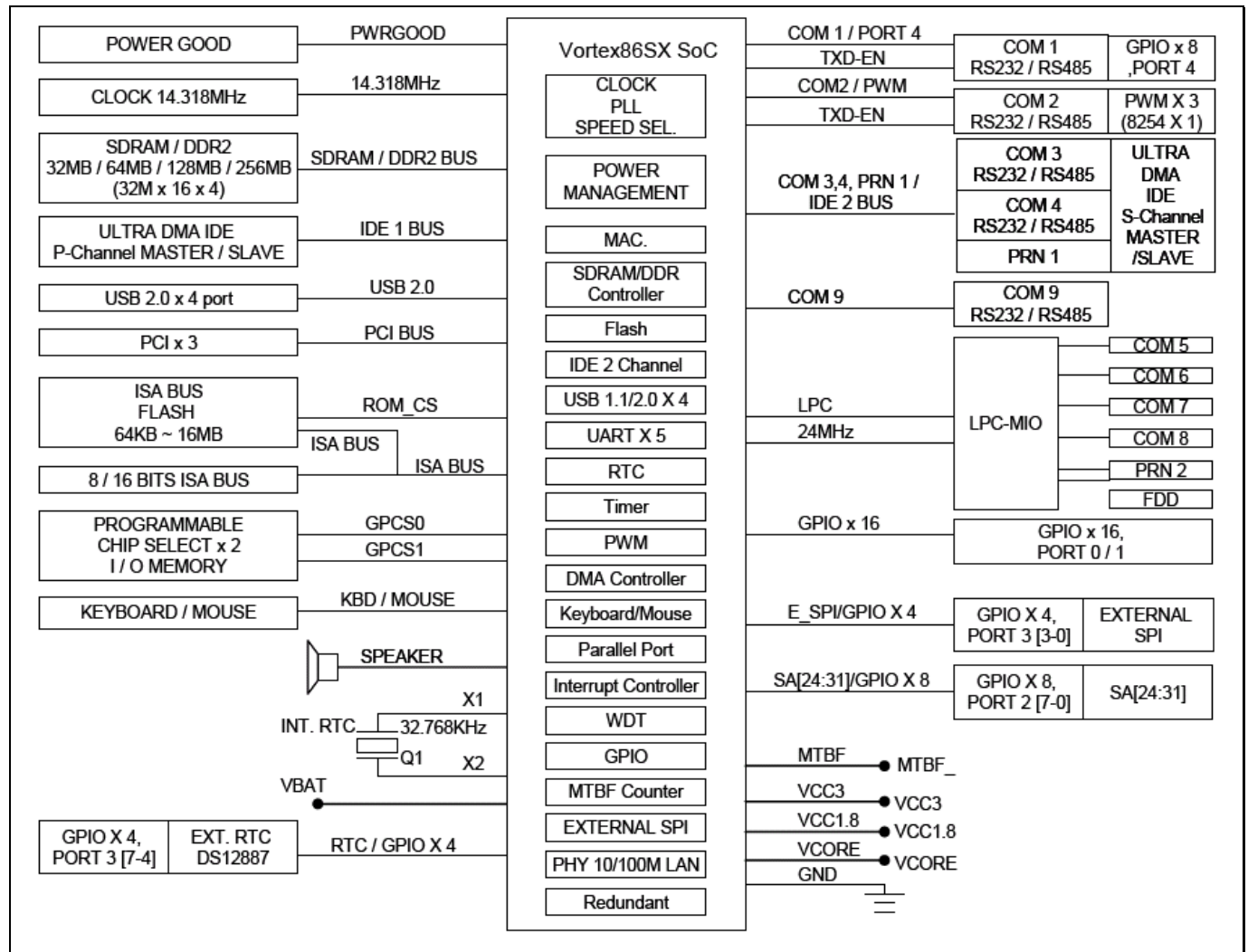


## Using GPIO in Vortex86SX SoC

2007-05-28

40 GPIO pins are provided by the Vortex86SX for general usage in the system. All GPIO pins are independent and can be configured as inputs or outputs, with or without pull-up/pull-down resistors. Refer to Vortex86SX functions block diagram:



GPIO port 0,1 and 2 are always free for use normally. If your system does not use external RTC and SPI, GPIO port 3 is also free for use. Developer also can disable COM1 to select GPIO port 4. The actual free GPIO pins depend on your system. Please check it before using GPIO.

## Setup GPIO Direction

Here is GPIO direction and data registers:

	Port 0	Port 1	Port 2	Port 3	Port 4	Description
<b>Data Register</b>	78H	79H	7AH	7BH	7CH	
<b>Direction Register</b>	98H	99H	9AH	9BH	9CH	0: GPIO pin is input mode 1: GPIO pin is output mode

If send value 0FH to port 98H, it means that GPIO port0 [7-4] are input mode and port[3-0] are output mode.

If send value 00H to port 98H, it means that GPIO port0 [7-0] are input mode.

If send value FFH to port 98H, it means that GPIO port0 [7-0] are output mode.

If send value 03H to port 98H, it means that GPIO port0 [7-2] are input mode and port[1-0] are output mode.

## DOS Example

```
#include <stdio.h>

void main(void)
{
    /* set GPIO port0[7-0] as input mode */
    outportb(0x98, 0x00);

    /* read data from GPIO port0 */
    inportb(0x78);

    /* set GPIO port1[7-0] as output mode */
    outportb(0x99, 0xff);

    /* write data to GPIO port1 */
    outportb(0x79, 0x55);

    /* set GPIO port2[7-4] as output and [3-0] as input*/
    outportb(0x9a, 0xf0);

    /* write data to GPIO port2[7-4], the low nibble (0x0a) will be ignored */
    outportb(0x7a, 0x5a);

    /* read data from port2[3-0] */
    unsigned char c = inportb(0x7a) & 0x0f;

    /*--- if GPIO port3 is free, those codes can work ---*/

    /* set GPIO port3[7-2] as output and [1-0] as input*/
    outportb(0x9b, 0xfc);

    /* write data to GPIO port2[7-2], the bit 1-0 will be ignored */
    outportb(0x7b, 0xa5);

    /* read data from port3[1-0] */
    unsigned char c = inportb(0x7b) & 0x03;

    /*--- if GPIO port4 is free, those codes can work ---*/

    /* set GPIO port4[7,5,3,1] as output and port4[6,4,2,0] as input*/
```

```

outportb(0x9c, 0xaa);

/* write data to GPIO port4[7,5,3,1], the bit 6,4,2 and 0 will be ignored */
outportb(0x7c, 0xff);

/* read data from port4[6,4,2,0] */
unsigned char c = inportb(0x7c) & 0xaa;
}

```

## Linux Example

```

#include <stdio.h>

#include <stdio.h>
#include <sys/io.h>
#define outportb(a,b) outb(b,a)
#define inportb(a) inb(a)

void main(void)
{
    iopl(3);

    /* set GPIO port0[7-0] as input mode */
    outportb(0x98, 0x00);

    /* read data from GPIO port0 */
    inportb(0x78);

    /* set GPIO port1[7-0] as output mode */
    outportb(0x99, 0xff);

    /* write data to GPIO port1 */
    outportb(0x79, 0x55);

    /* set GPIO port2[7-4] as output and [3-0] as input*/
    outportb(0x9a, 0xf0);

    /* write data to GPIO port2[7-4], the low nibble (0x0a) will be ignored */
    outportb(0x7a, 0x5a);

    /* read data from port2[3-0] */
    unsigned char c = inportb(0x7a) & 0x0f;

    /*--- if GPIO port3 is free, those codes can work ---*/

    /* set GPIO port3[7-2] as output and [1-0] as input*/
    outportb(0x9b, 0xfc);

    /* write data to GPIO port2[7-2], the bit 1-0 will be ignored */
    outportb(0x7b, 0xa5);

    /* read data from port3[1-0] */
    unsigned char c = inportb(0x7b) & 0x03;

    /*--- if GPIO port4 is free, those codes can work ---*/

    /* set GPIO port4[7,5,3,1] as output and port4[6,4,2,0] as input*/
    outportb(0x9c, 0xaa);

    /* write data to GPIO port4[7,5,3,1], the bit 6,4,2 and 0 will be ignored */

```

```
outportb(0x7c, 0xff);

/* read data from port4[6,4,2,0] */
unsigned char c = inportb(0x7c) & 0xaa;
}
```

## Windows CE Example

```
#include "stdafx.h"

unsigned char inportb(int addr)
{
    __asm
    {
        push edx
        mov edx, DWORD PTR addr
        in al, dx
        and eax, 0xff
        pop edx
    }
}

void outportb(int addr, unsigned char val)
{
    __asm
    {
        push edx
        mov edx, DWORD PTR addr
        mov al, BYTE PTR val
        out dx, al
        pop edx
    }
}

void main(void)
{
    /* set GPIO port0[7-0] as input mode */
    outportb(0x98, 0x00);

    /* read data from GPIO port0 */
    inportb(0x78);

    /* set GPIO port1[7-0] as output mode */
    outportb(0x99, 0xff);

    /* write data to GPIO port1 */
    outportb(0x79, 0x55);

    /* set GPIO port2[7-4] as output and [3-0] as input*/
    outportb(0x9a, 0xf0);

    /* write data to GPIO port2[7-4], the low nibble (0x0a) will be ignored */
    outportb(0x7a, 0x5a);

    /* read data from port2[3-0] */
    unsigned char c = inportb(0x7a) & 0x0f;

    /*--- if GPIO port3 is free, those codes can work ---*/

    /* set GPIO port3[7-2] as output and [1-0] as input*/
```

```
outportb(0x9b, 0xfc);

/* write data to GPIO port2[7-2], the bit 1-0 will be ignored */
outportb(0x7b, 0xa5);

/* read data from port3[1-0] */
unsigned char c = inportb(0x7b) & 0x03;

/*--- if GPIO port4 is free, those codes can work ---*/

/* set GPIO port4[7,5,3,1] as output and port4[6,4,2,0] as input*/
outportb(0x9c, 0xaa);

/* write data to GPIO port4[7,5,3,1], the bit 6,4,2 and0 will be ignored */
outportb(0x7c, 0xff);

/* read data from port4[6,4,2,0] */
unsigned char c = inportb(0x7c) & 0xaa;
}
```

## Technical Support

For more technical support, please visit <http://www.dmp.com.tw/tech> or mail to [tech@dmp.com.tw](mailto:tech@dmp.com.tw).