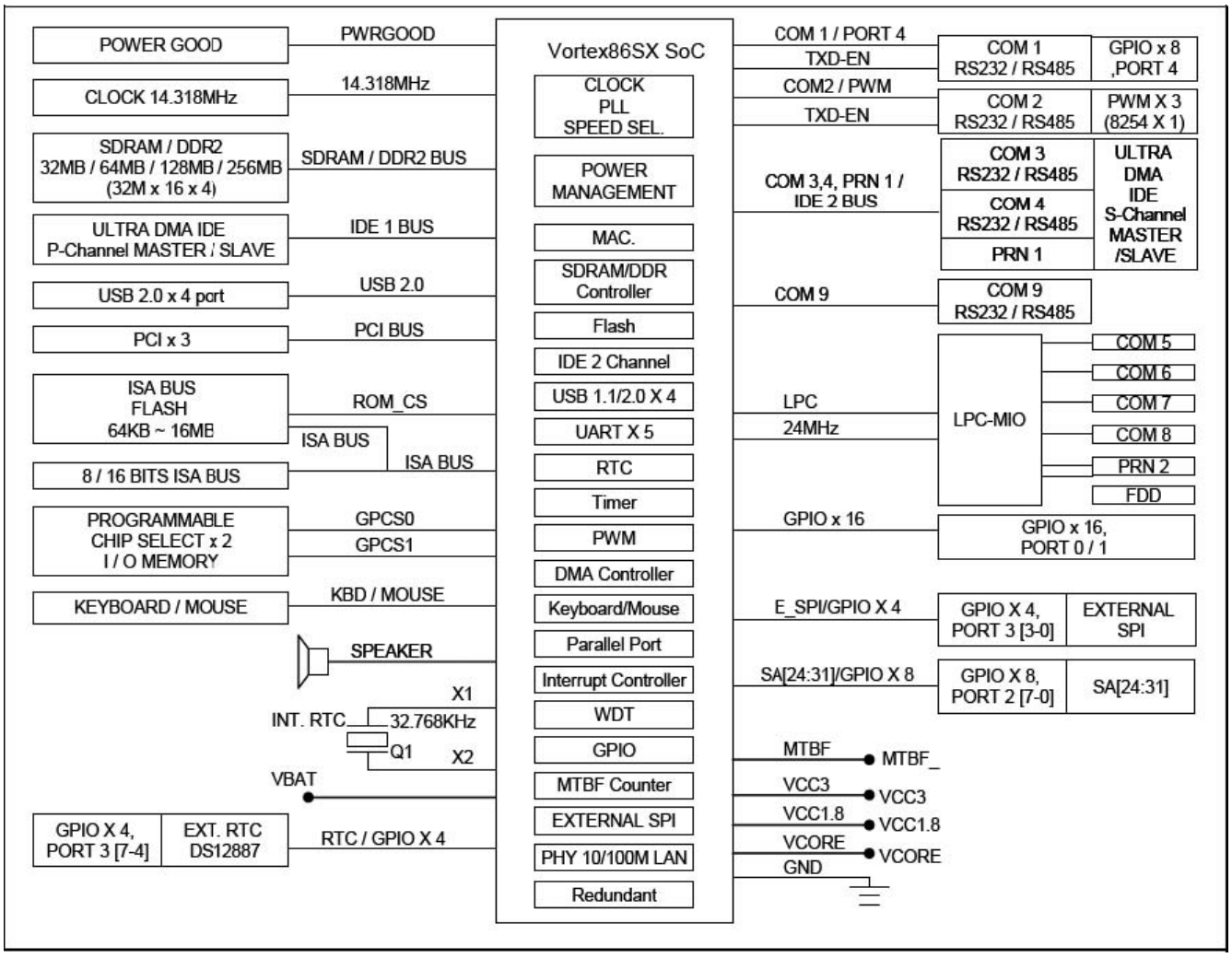
Usage of GPIO with RB-262SX (Vortex86SX SoC)

Content

1. Connection description

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.

2. Physical connection on RB-262SX

Totally there is available 4 GPIO ports, each with 8-bit input/output.

First three ports are provided are provided through box header on front panel through 26-pin connector (2 row, 2.54mm).
Additionally there is possibility to use last 8-bit GPIO shared on COM1 port pin. When COM1 setup into GPIO mode you cannot use RS-232.

Pin summary:

Signals	GPIO P00-P07	GPIO P10-P17	GPIO P20-P27	GPIO P40-P47	PWM** P0-P2
Connector	GPIO 26pin header			COM1*	COM2*

Please note COM1 must be setup into GPIO mode in BIOS Setup. If not available in BIOS please upgrade BIOS into A3 version. When COM1 port setup into GPIO mode then RS-232 signals does not working, and vice versa.
****PWM features accessible through COM2 not described in this document.***

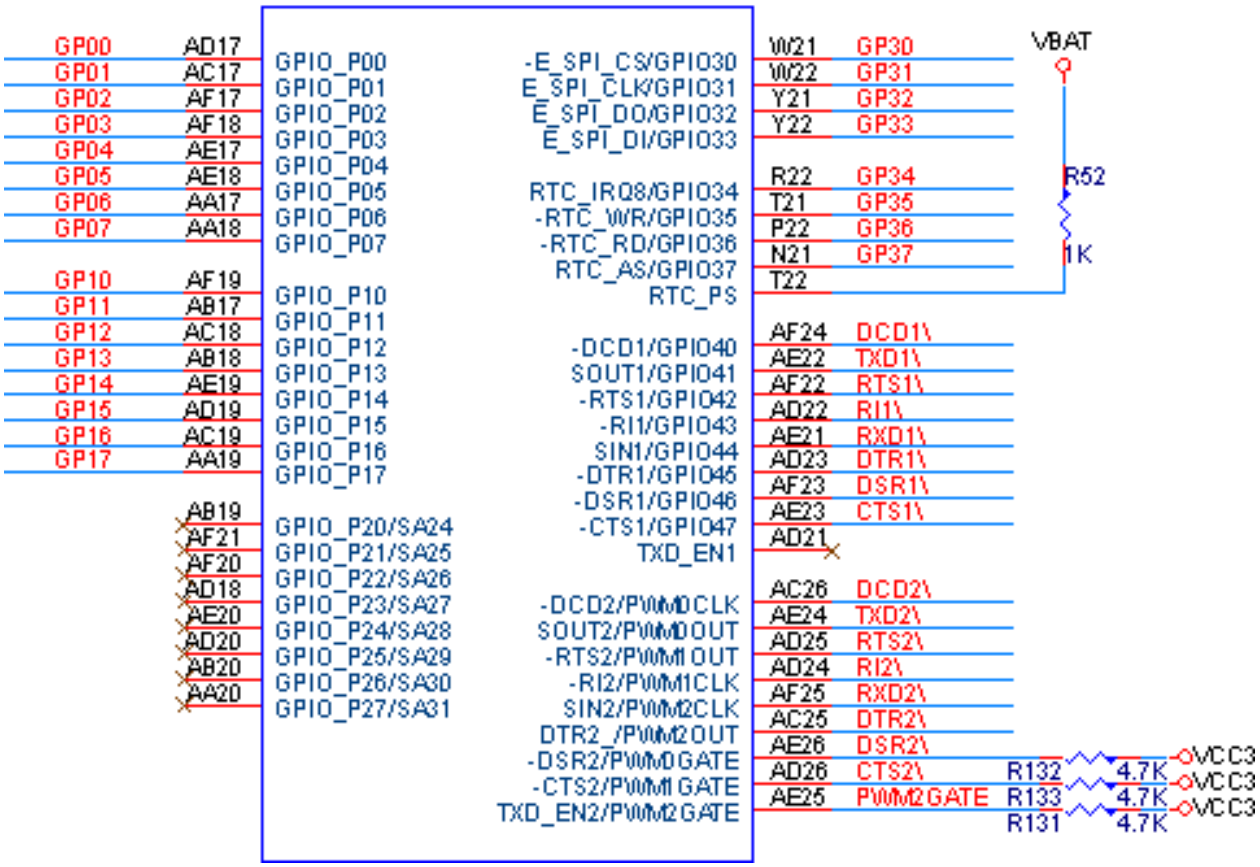
Pin description, box header on front panel
26-pin, 2.54mm, 2 row

Pin#	**Signal Name**	**Pin#**	**Signal Name**
1	VCC	2	GND
3	GP10	4	GP00
5	GP11	6	GP01
7	GP12	8	GP02
9	GP13	10	GP03
11	GP14	12	GP04
13	GP15	14	GP05
15	GP16	16	GP06
17	GP17	18	GP07
19	GP34	20	GP30
21	GP35	22	GP31
23	GP36	24	GP32
26	GP37	26	GP33

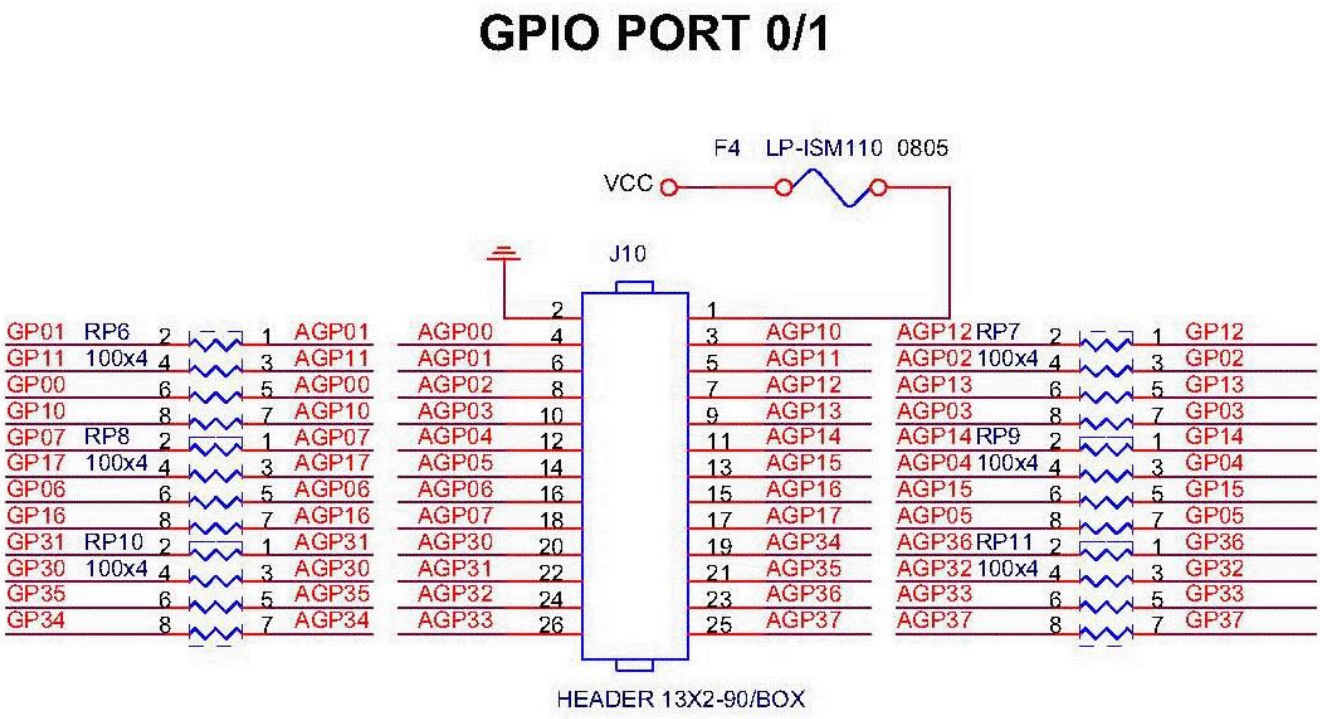
COM1 - 9-pin D-sub Connector

Pin#	Signal Name	Pin #	Signal Name
1	DCD1/GP40	2	RXD1/GP44
3	TXD1/GP41	4	DTR1/GP45
5	GND	6	DSR1/GP46
7	RTS1/GP42	8	CTS1/GP47
9	RI1/GP43	--	--

Internal connection circuit scheme



Header connection scheme



VCC GPIO notice

User can feed devices connected on GPIO port through pin VCC. However there should be considered their limitations.
VCC pin has in circuit PTC fuse (LP-ISM110) which limits current from this output. Maximal continuous current is **1,1A (5,5W)**. For short period is allowed to feed 1,3A. When device feeds more than 1,3A then PTC device starts to trip into high impedance state. Trip time is about 0,3s @ 8A. Maximal current which can PTC withstand without damage is 40A.

Please note - own consumption of RB-262SX is about 0,6A without any other peripherals (users case differ on installed peripherals including keyboard). Power adaptor supplied with RB-262SX has output 2A. In default configuration there is about 1,4 A available from default power adaptor to feed devices connected on GPIO.

3. Setup GPIO Direction

Here is GPIO direction and data registers:

	Port 0	**Port 1**	**Port 2**	**Port 3**	**Port 4**	**Description**
Data Register	78H	79H	7AH	7BH	7CH	
Direction Register	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.

4. 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 and0 will be ignored */
outportb(0x7c, 0xff);
/* read data from port4[6,4,2,0] */
unsigned char c = inportb(0x7c) & 0xaa;
}
```

5. 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 and0 will be ignored */
outportb(0x7c, 0xff);
/* read data from port4[6,4,2,0] */
unsigned char c = inportb(0x7c) & 0xaa;
}
```

6. 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;
}
```