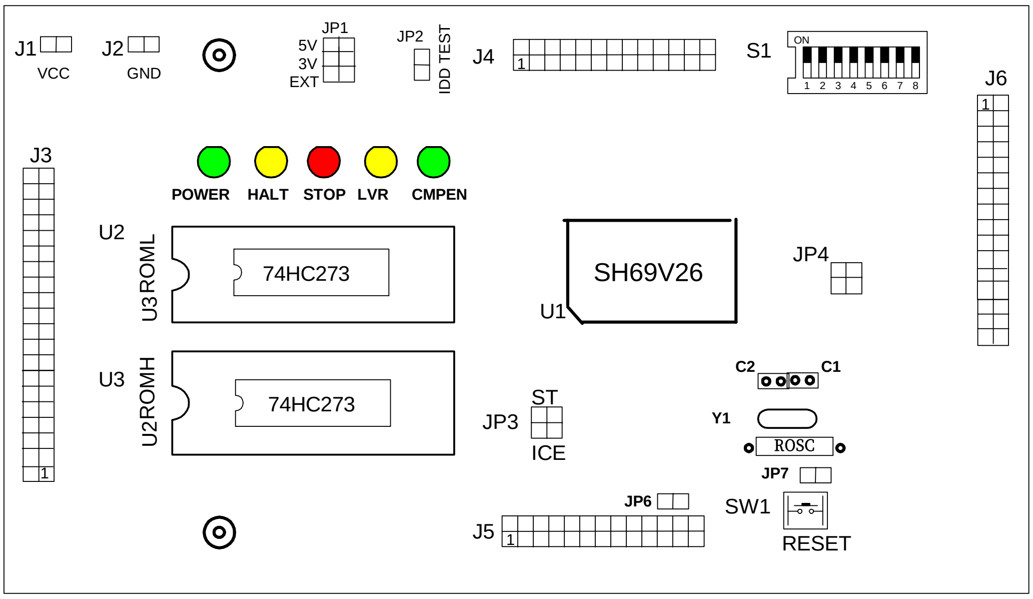
SH69P26 EVB

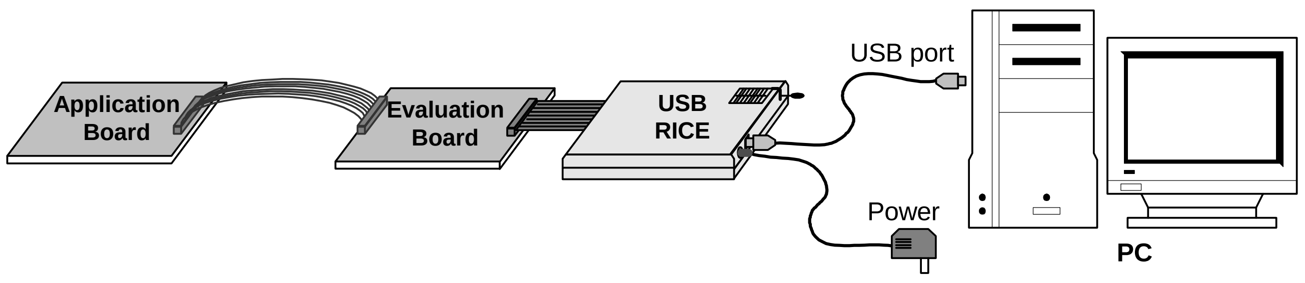
Application Note for SH69P26 EVB

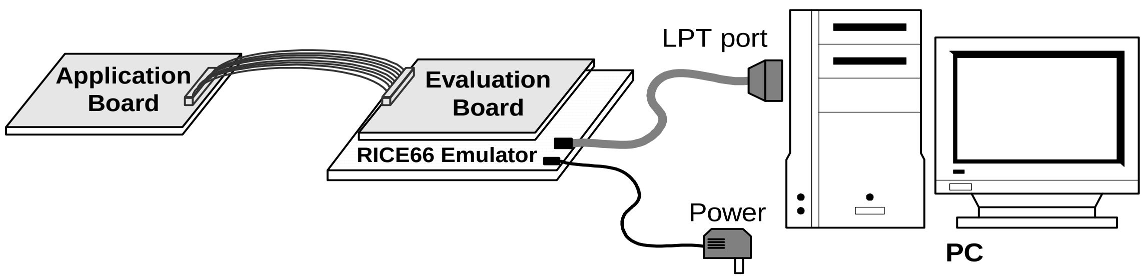
SH69P26 EVB

The SH69P26 EVB is used to evaluate the SH69P26 chip's function for the development of application program. It contains an SH69V26 chip for evaluating the functions of SH69P26. The following figure shows the placement diagram of SH69P26 EVB.



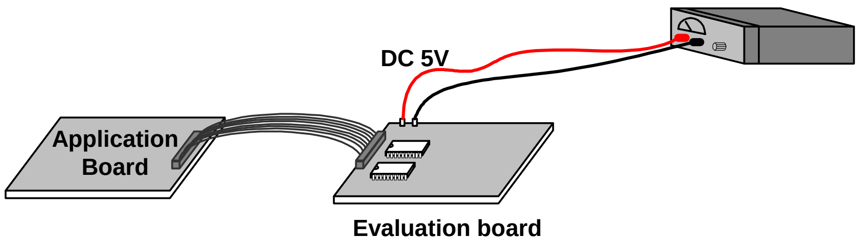
There are two configurations of SH69P26 EVB in application development: ICE mode and stand-alone mode.
In ICE mode, the ICE (motherboard) is connected to the EVB by ICE interface.





(a) ICE mode

In standalone mode, the EVB is no longer connected to the motherboard, but the Flash (or EPROM) must be inserted to the socket that stored the application program.



(b) Stand-alone mode

The process of your program's evaluation on SH69P26 EVB

User can use **Sino Wealth Rice66 Integrated Development Environment (IDE)** to emulate the program and produce the obj file. Rice66 IDE is a real-time in-circuit emulator program. It provides real-time and transparent emulation support for the SH6X series 4-bit microcontroller. And integrate assembler can create binary (*.obj) file and the other files.

Use Flash (or EPROM) In standalone mode

Rice66 IDE is built-in with an object file depart function. The command "Split object file" can separate the one 16 bits object file into two 8 bits files, which contain the high and low bytes respectively.

Write the high/low byte obj file to Flash (or EPROM) and insert them to EVB (ROMH and ROML). Then, user can evaluate the program in standalone mode.

SH69P26 EVB Interface Connector: (Top View from EVB)

□ **Port interface connector: J6 (TOP View from EVB)：**

PG2	PG1
PG3	PG0
PE3	PF3
PE2	PF2
PE1	PF1
PE0	PF0
PC2	PC1
PC3	PC0
GND	VCC
PA0	PB3
PA1	PB2
PA2	PB1
PA3	PB0
PD0	PD3
PD1	PD2
PH0	PH1

□ **External VCC input for stand alone mode:**

J1, J2 -The external power input when the EVB worked in stand-alone mode. The voltage of Vcc must be 5V±5%.

□ **Interface to ICE:**

J4, J5 -connect to RICE66 2.0, or connect to RICE66 3.0 with a transition board.

J3 -connect to RICE66 3.0 directly.

□ **Interface to test the EV chip operating current**

JP2 - User can test the EV chip current through JP2

Note: In ICE mode, the current value is correct only when the RICE66 runs in external clock from EVB mode. (Select the "external clock from EVB" in OSC Frequency Config manual.)

Switch setting:

S1

Bit 8	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Remarks**
RST	LVR0	LVR	WDT	OSC3	OSC2	OSC1	OSC0	
X	X	X	X	X	Off	Off	Off	External clock
X	X	X	X	X	Off	Off	On	Internal ROSC RC oscillator 2M
X	X	X	X	X	Off	On	Off	Internal ROSC RC oscillator 4M
X	X	X	X	X	Off	On	On	Internal ROSC RC oscillator 6M
X	X	X	X	X	On	Off	Off	External ROSC RC oscillator
X	X	X	X	X	On	Off	On	Ceramic resonator
X	X	X	X	X	On	On	Off	Crystal oscillator
X	X	X	X	X	On	On	On	32.768KHz Crystal oscillator
X	X	X	X	Off	X	X	X	2 ~ 8MHz
X	X	X	X	On	X	X	X	400KHz ~ 2MHz
X	X	X	Off	X	X	X	X	WDT Enable
X	X	X	On	X	X	X	X	WDT Disable
X	X	Off	X	X	X	X	X	LVR Disable
X	X	On	X	X	X	X	X	LVR Enable
X	Off	X	X	X	X	X	X	High LVR voltage
X	On	X	X	X	X	X	X	Low LVR voltage
Off	X	X	X	X	X	X	X	Reset pin Enable
On	X	X	X	X	X	X	X	Reset pin Disable

#Note : LVR value might exceed the SPEC range.

Jumper setting:

JP1	EV chip power supply select
Short at 3V position	The power of EV chip is set as internal 3V power source.
Short at 5V position	The power of EV chip is set as internal 5V power source.
Short at EXT position	The EV chip use external power supply that was input from EXT_VDD pin.

JP3	EVB ICE/Stand-alone mode select
Short at Stand-alone position	Select stand-alone mode. (The system clock is provided by the on board oscillator.)
Short at With-ICE position	Select with-ICE mode. (The system clock is provided by the ICE.)

JP4	OSCI OSCO share select
Short at OSCI position	OSCI pin will be disabled and shared as PORTC0
Short at OSCO position	OSCO pin will be disabled and shared as PORTC1.

JP6	STACK overflow select
Short	The stack overflow function in the ICE mode will on
Open	The stack overflow function in the ICE mode will off

JP7	RESET pin share select*
Short	RESET pin will be disabled and shared as PORTC.3
Open	RESET pin will be enabled.

*Note: If external Reset pin is enabled (PORTC.3 is shared as Reset pin) in main chip application, SH69P26 will provide better performance on Electro Magnetic Compatibility (EMC)

SW1 (RESET):

Reset the whole system by pressing the button.

Diagnostic LED:

Power LED:	The LED will be turned on when the EVB is powered.
HALT LED:	The LED will be turned on when the system is in HALT mode.
STOP LED:	The LED will be turned on when the system is in STOP mode.
LVR LED:	The LED will be turned on during the low voltage reset active.
CMP LED:	The LED will be turned on when CMP is active.

Notes:

Application notes:

1.1 After entering into the RICE66 and successfully downloaded the user program, use the F5 key on the PC keyboard to reset the EVB before running the program. If abnormal response occurs, the user must switch off the ICE power and quit RICE66, then wait for a few seconds before restarting.

1.2 When running the RICE66 for the first time, the user needs to select the correct MCU type, clock frequency ... then save the settings and restart RICE66 again.

1.3 Can't Step (F8) or Over (F9) a HALT and STOP instruction.

1.4 Can't emulate the interrupt function in Step (F8) operating mode.

1.5 When you want to escape from HALT or STOP (in ICE mode), please press F5 key on the PC keyboard twice.

1.6 The maximum current limit supplied from EVB to the target is 100mA. When the current in the target is over 100mA, please use external power supply.

1.7 When EV board worked in "with ICE" mode, user can use the clock from EXOSC_IN (JP7) as the system clock. (Refer to the RICE66 User's Guide).

Programming notes:

2.1 Clear the data RAM and initialize all system registers during the initial programming.

2.2 Never use the reserved registers.

2.3 Do not execute arithmetic operation with those registers that only have 1, 2 or 3 bits. This kind of operation may not produce the result you expected.

2.4 To add "p=69P26" and "romsize=6144" at the beginning of a program. If any problem occurs during the compilation of the program, check the device and set if it was set correctly.

2.5 Both index register DPH and DPM have three bits; so pay attention to the destination address when using them.

2.6 The clock of the Base timer and LCD driver is sourced by the low frequency oscillator (OSC). When evaluating the LCD and the Base-timer function, the "External from EVB" item in the Config manual should be selected.

2.7 The capacitors for LCD power supply must be connected at all time, whether the LCD is being used or not. It takes more than 0.3 second to wake up from STOP mode when using 32.768kHz crystal. So, if the system is awake when the key is pressed, the key may have been released when the program starts to read the Key value.

2.8 Notes for interrupt:

2.8.1 Please make sure that the IE flag is enabled before entering into a "HALT" or a "STOP mode. It means that the "HALT" or "STOP" instruction must follow the set "IE" instruction closely.

2.8.2 After the CPU had responded to an interrupt, IRQ should be cleared before resetting IE in order to avoid multi-responses.

2.8.3 Interrupt Enable instruction will be automatically cleared after entering into the interrupt-processing subroutine. If setting IE is too early, it is possible to reenter into the interrupt. So the Interrupt Enable instruction should be placed at the last 3 instructions of the subroutine.

2.8.4 CPU will not respond to any interrupt during the next two instructions after the Interrupt Enable flag be set from 0 to 1.

2.8.5 After CPU has responded to an interrupt, IE will be cleared by the hardware. It is recommended to clear the IRQ at the end of interrupt subroutine.

2.8.6 The stack has four levels. If an interrupt is enabled, there will be only three levels that can be used.

2.8.7 It is recommended that the last line of program is "END".

Examples:

1> Description: CPU can not wakeup after executing the "HALT" or "STOP" instruction.

Program: Interrupt Enable instruction is set outside the interrupt subroutine

<Wrong example>	**<Correct example >**
……	……
LDI IE, 0FH ; enable interrupt	LDI IE, 0FH ; enable interrupt
NOP	~~NOP~~
NOP	~~NOP~~
HALT	HALT

Analysis: After two "NOP" instructions, if an interrupt request comes or IRQ is non-zero during the third instruction cycle, CPU will respond to the interrupt and IE will be cleared. Then when returning to main program, CPU starts to execute "HALT" or "STOP" and will not be activated, because IE is cleared to zero and all interrupts are disabled.

Solution: "HALT" or "STOP" are being followed closely by the "LDI IE, 0FH"

2> Description: CPU responds to one interrupt several times.

Program: Interrupt Enable instruction is placed outside the interrupt subroutine.

```
L1:
……
LDI     IE, 0FH         ; enable interrupts
NOP
NOP
JUMP   L1
```

Analysis: After executing this two "NOP" instructions, and IRQ is not cleared in time, CPU will respond to the interrupt again when it executes the two instructions followed by "LDI IE, 0FH". This will happen again and again. So CPU responds to one interrupt several times.

Solution: The relative IRQ flag is cleared in time after responding to the interrupt.

3> Description: CPU is running dead in the interrupt-processing program.

Program: an interrupt subroutine.

```
ENTERINT:
    ……
    LDI        IE, 0FH
    NOP
    LDA STACK, 0
    RTNI
```

Analysis: After executing “LDI IE, 0FH” and the following two instructions, an interrupt request comes or the last relative IRQ flag is not cleared in time, then CPU will respond to the interrupt again, so the interrupt is nesting again. When the stack is over 4 levels, it will run into a dead loop.

Solution: Make sure that the CPU can quit from interrupt subroutine within two instruction cycles after interrupt is enabled; After the interrupt is responded, the relative IRQ flag should be cleared before enabling the interrupt.

2.9 Notes for TIMER

2.9.1 When setting the Timer Counter, write first T0L, then T0H.

2.9.2 After setting TM0, T0L, T0H, there is no need to rewrite after the Timer counts overflow, otherwise it will cause a time error every time. The timer is interrupted by the reload registrar that was set in different time.

2.10 Notes for I/O

2.10.1 Never do the logical operation with the I/O ports. Especially when the I/O ports are connected with the other components externally.

2.10.2 When the internal pull-high resistor is turned on, “1” must be written to I/O Port before Reading.

2.10.3 When counting external pulse, please directly read the PORT status to make sure that the counted number is correct.

2.10.4 The Key De-bounce time is recommended to be 50ms. But in the Rubber Key application, it is best to test Rubber Key’s De-bounce time.

2.11 When the Compiler of old version compiles program, the last line will be read twice. So, if the last line is an instruction, two same operations will be occurred. If there is Label in the last line, compiler will give an error named ‘repeated definition ’ . This will happen in main program or included files and it is recommended that the last line should be a blank line or END.

2.12 It takes 0.3 second to wake up from STOP when using 32768Hz crystal. So, if the system is waked up by key pressing, the key may have been released when the program begins to read Key value. Please pay more attention to this problem.

2.13 **In 28 pin mode, All bits of the $22 RAM are reserved, Always keep it to “0” in the User’s program.**

2.14 **In 28 pin mode, All bits of the $20 RAM are reserved, Always keep it to “1” in the User’s program.**

Application notes Revision History

Revision No.	History	Date
0.1	Add reset pin share function	Jan.2006
0.0	Original	Oct.2005