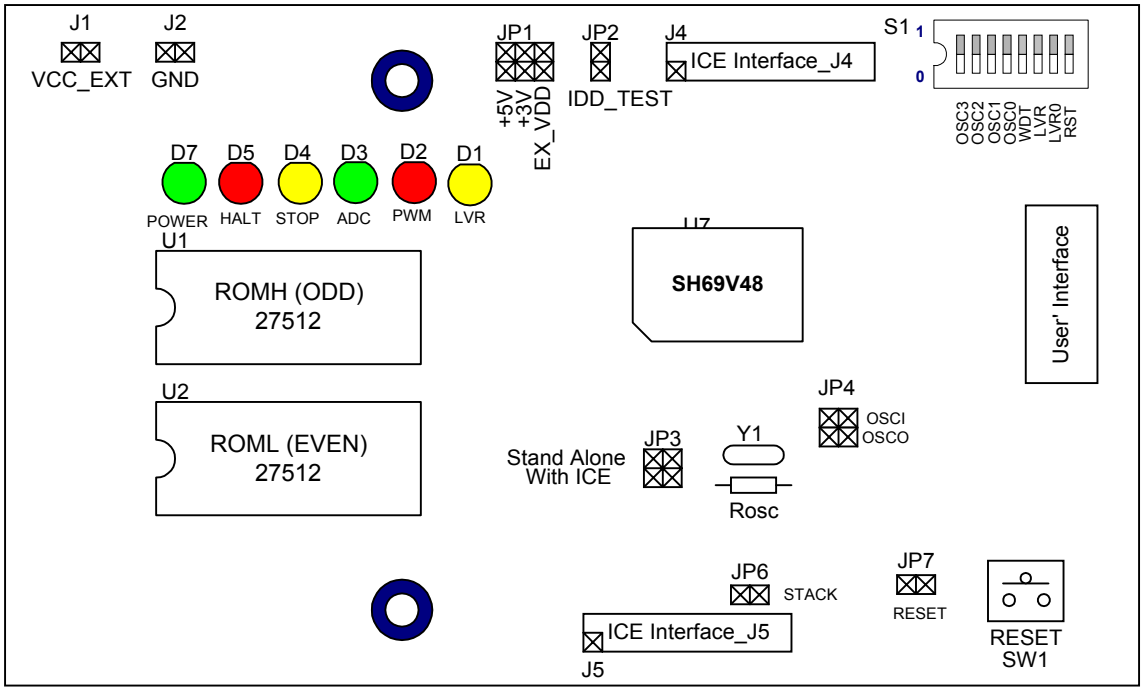
SINO WEALTH

SH69P48 EVB

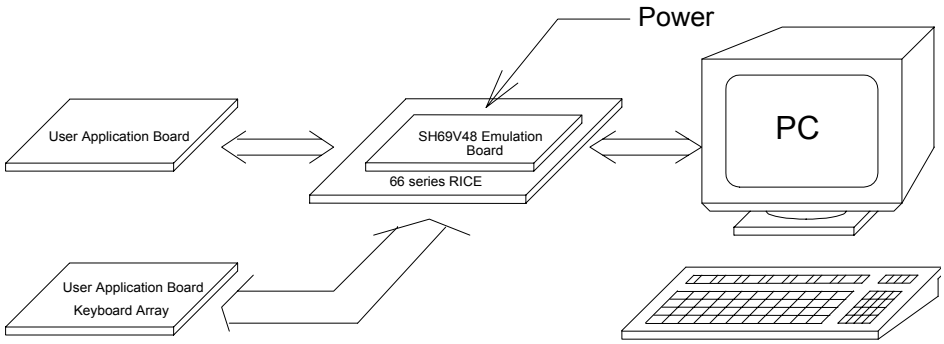
Application Notes for SH69P48 EVB

SH69P48 EVB

The SH69P48 EVB is used to evaluate the SH69P48 chip's function for the development of application program. It contains of a SH69V48 chip to evaluate the functions of SH69P48 including the ADC input and the PWM output. The following figure shows the placement diagram of SH69P48 EVB.

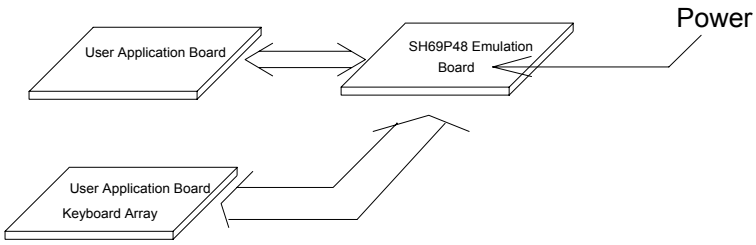


There are two configurations of SH69P48 EVB in application development: ICE mode and stand-alone mode.
In the ICE mode, the SH66xx ICE (motherboard) is connected to the SH69P48 EVB by the ICE interface.



(a) ICE mode

In the standalone mode, the SH69P48 EVB is no longer connected to the motherboard. But the EPROM chip has must be connected to EPROM sockets of the SH69P48 EVB. The EPROM stores the application program; they may be the 27512.



(b) Stand-alone mode

The process of your program's evaluation on SH69P48 EVB

Uasm66.exe: assemble the program, and get binary (*.obj) file and the other files.
Depart the one 16 bits obj file to the two 8 bit files by convert.exe.

Usage example (for example: aaa.asm):

1.Run the SH66 series assemble program:
C: >uasm66 aaa.asm ; to produce the obj file: aaa.obj
2. Depart the aaa.obj to two 8-bit file aaah.obj and aaal.obj, for example:
C: > convert
Input the 16 bits (.obj) file aaa.obj
Then aaah.obj and aaal.obj will be created.
3. Write the aaah.obj to EPROM (ROMH)
Write the aaal.obj to EPROM (ROML)
4. Put the two EPROM (ROML and ROMH) into the EPROM sockets.

SH69P48 EVB Programming Notices:

1. Refer to ADC notes in the SH69P48 datasheet.
2. Refer to PWM notes in the SH69P48 datasheet.
3. Clear data RAM and initialize all system registers at the beginning.
4. Do not perform logical operation with I/O ports. Especially when the I/O ports have external connections.
5. Do not perform arithmetic operation with those registers only have 1, 2 or 3 bit. This kind of operation may not get the result you expected.
6. Never use reserved registers.
7. If “IE” instruction (interrupt enable) is set outside the interrupt processing program and there is “HALT” or “STOP” instruction, this two instructions should be followed “IE” instruction closely.
8. After CPU responding to an interrupt, IRQ should be cleared before resetting IE in order to avoid many responses to one interrupt.
9. Interrupt Enable instruction will be automatically cleared after entering interrupt-processing program. If setting IE too early, there is a possibility of re-entry the interrupt. So the Interrupt Enable instruction should be placed at the end and followed closely by two instructions include “RTNI”.
10. During the two successive instruction cycles next to Interrupt Enable instruction, CPU will not respond to any interrupts.
11. After CPU responding to interrupts, each bit of IE will be cleared by hardware while IRQ should be cleared by software.
12. It is necessary to add NOP before or after the HALT instruction, else the CPU will execute error instruction when it wakes up from the HALT.

```
.
. .
NOP
HALT
NOP
.
. .
```

13. It is wise to set Interrupt Enable flag before you return from subroutine in two instructions.

```
.
. .
LDI   IE, 04H  ; Enable timer0 interrupt
LDA    Temp, 0
RTNI
```

14. When you set interrupt enable flag as the following and your subroutine do not set Interrupt Enable flag, then your system will never wake up if an interrupt entered between the NOP.

Wrong Example:

```
.
. .
LDI    IE, 1111B;  IE = Interrupt enable flag
NOP
NOP
NOP
HALT
.
. .
```

15. To add "p=69P48" and "romsize=4096" at the beginning of a program. If problem is found in compiling the program. Check if the SH6566.dev is located at the program directory.
16. When setting Timer Counter, first fill T0L/T1L, then T0H/T1H.
17. After setting TM0/TM1, T0L/T1L, T0H/T1H, it is unnecessary to reset them after interrupt each time. If TM0/TM1, T0L/T1L, T0H/T1H are reset after each TIMER interrupt, interrupt interval time will not equal because the interrupt timing is not successive.
18. Any instruction containing writing to or reading from memory, it should not be used to operate with I/O Port. It is best to avoid using those instructions such as “SUB, ADD “ which do not contain Write operation with I/O Port but have computation operation.
19. When the Port of SH69P48 is used as Key Scanning mode, writing it and reading it should be separated by 2 or 3 NOP.
20. “1” must be written to I/O Port before Reading.
21. Writing “1” to I/O Open Drain and then entering “STOP” will cause current leakage ranging from tens to hundreds micro-ampere. So pull-up or pull-low resistors value from 1 to 2 MΩ must be used to prevent I/O Float when I/O in Open Drain mode.
22. Directly reading PORT states ensure the count is correct.
23. Interrupt activating from STOP at the first time can save power.

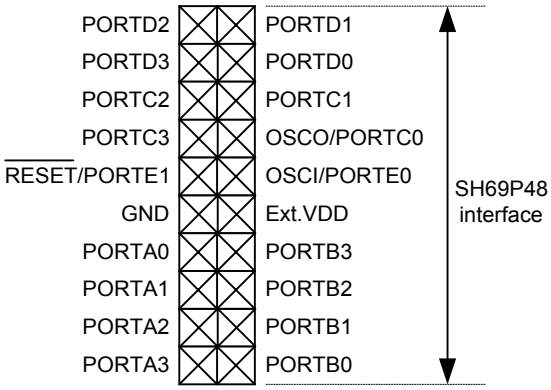
24. When the Compiler of old version compiles program, the last line will be read twice. So, if the last line is an instruction, two same operations will be occurred. If there is Label in the last line, compiler will give an error named 'repeated definition' . This will happen in main program or included files and it is recommended that the last line should be a blank line or END.
25. The stack has eight layers, if an interrupt is enabled, there only have seven layers can be used. Or else, if an interrupt comes, the stack will be overflowed that will cause CPU Reset or other errors.
26. Key De-bounce time is recommended to be 50ms. If a user use Rubber Key, it is best to test Rubber Key's De-bounce time.
27. It takes 0.8 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.
28. Index register DPH and DPM both have three bits only, so pay attention to the referred address when using them.
29. The "NOP" instruction should be added at the beginning of the program to ensure the IC is stability.

SH69P48 interface connector: (Top View from EVB)

J1, J2 (VCC_EXT, GND):

- External Power input for Stand-Alone mode. The voltage of VCC_EXT must be 5V±5%.

J6 (User's interface):



J3 (With ICE interface):

- Connect with RICE66.

Jumper setting:

JP1 (SH69V48 chip power select):

- If short the +5V positions, the voltage (+5V) of SH69V48 is internal source. (Default)
- If short the +3V positions, the voltage (+3V) of SH69V48 is internal source.
- If short the EX_VDD positions, you can external input the power (2.4 ~ 5.5V, refer to SH69P48 spec.) of SH69V48 from Ext.VDD.

JP2 (SH69V48 chip power import):

- This jumper must be connecting. (Default)
- If you want to get the operating current (for reference only), remove the jump and link a current-meter.

JP3 (SH69P48 EVB ICE/Stand-alone mode select):

- If short the "With ICE" positions, the clock of SH69P48 EVB is fed from ICE. This is only for ICE mode.
- If short the "Stand alone" positions, the clock of SH69V48 is provided from its oscillator (refer to SH69P48spec.). This is only for stand-alone mode.

JP4 (OSCO/PORTC0, OSCI/PORTE0 port type select):

- If select "Internal RC oscillator", short OSCI end and OSCO end of JP4, PORTC0 and PORTE0 can be acquired.
- If select "External RC oscillator", short OSCO end of JP4, PORTC0 can be acquired.
- If select "Crystal oscillator" or "Ceramic resonator", do not short any end.

JP6 (STACK overflow ON/OFF):

- The stack overflow function in ICE mode will on when it is short. (Default)

JP8, JP9 (Package select):

- If select 20 pin package, short the 'GND' end of the JP8 and JP9.
- If select 16 pin package, short the 'GND' end of the JP8 and the 'VDD1' end of the JP9.
- If select 8 pin package, short the 'VDD1' end of the JP8 and JP9.

Switch setting:

S1: (0: OFF, 1: ON)

- Bit0: OSC3 (Select Oscillator range)

  0 = 2 ~ 10MHz (Default)
  1 = 30KHz ~ 2MHz

- Bit1~3: OSC2~0 (Select Oscillator type, Stand-alone mode only)

000 = External clock (Default)
001,010,011 = Internal RC oscillator
100 = External RC oscillator
101 = Ceramic resonator
110 = Crystal oscillator
111 = 32.768KHz Crystal oscillator

- Bit4: WDT (Watch dog timer Enable/Disable)
  0 = Enable (Default)
  1 = Disable
- Bit5: LVRF (Low Voltage Reset Disable/Enable)
  0 = Disable (Default)
  1 = Enable
- Bit6: LVRV (Select LVR voltage Range)
  0 = High LVR voltage (Default)
  1 = Low LVR voltage
- Bit7: Reset pin enable/disable
  0 = Enable (Default)
  1 = Disable

SW1 (RESET):

- Reset the EVB when push the button.

LED indication:

Low Voltage Reset indicate:
D1 is lighted during the low voltage reset active stage.

PWM indicate:
D2 is lighted when PWM is enabled

ADC module operate (ADCON=1) indicate:
D3 is lighted when ADC is operating.

STOP indicate:
D4 is lighted when the CPU was in STOP mode.

HALT indicate:
D5 is lighted when the CPU was in HALT mode.

Power indicate:
D7 is lighted when power is connected.

Notice:

Evaluate your program with ICE indicate:

1. After enter to RICE66 and successfully download the user program, push the F5 (Reset) on PC keyboard before run your program when you evaluate your program with ICE. If there were abnormal response, the user should power off the ICE, quit RICE66 and wait for a few seconds before restart.
2. First time run RICE66, need to select an appropriate MCU type, clock frequency ... save the settings and restart RICE66 again.
3. Can't execute Step (F8) or Step Over Call (F9) while the ICE worked in HALT and STOP instruction.
4. When you want to escape from HALT or STOP (in ICE mode), you should press the F5 key on PC keyboard twice.
5. The maximum current limit of the 3V power is 100mA, when the user uses internal 3V power to drive external device such as LED.
6. When the EVB worked in ICE mode, you can input the clock from EXOSC_IN as the system clock. (refer to the RICE66 ICE User' s Guide)