Programming tips

It may be useful to divide the register set to temporary data registers and permanent data registers, especially if some external event interrupts the processor regularly and interrupts are to be served with minimum delay. This way it is not necessary to store the contents of all the registers to the stack.

The *retu* command is used to return to the user mode.

It is advisable to write a simple service program for unused interrupts, for example only the return instruction *reti*, even if they would never be used. This insures that the unused interrupt registers (INTVi) hold the address of the return instruction. The best solution would, of course, be to write a small program for each unused interrupt, which generates an error message that an unused interrupt has been taken.

The returning from the interrupt is always done with the *reti* instruction.

The contents of conditional registers should always be stored to the memory in the interrupting program before the contents of the conditional registers are changed. This can be done by executing the *scon* instruction.

When returning after a system call back to a user program, the state of the processor is returned. (The contents of PSR_2 is copied to PSR) Normally this means that the processor returns to the mode it had before the system call was executed. The program execution is started at an address, which follows the system call by reading the contets of the SR31 register. It is not recommended to use the system call instruction *scall* in interrupt service routines, because it may result an unpredictable number of system calls, for example if system code is interrupted by a service program that also contains a system call. An exception occurs from two consecutive system calls. If it is necessary to serve an interrupt in a superuser mode, it can be done by setting the corresponding bit in the INT_MOD register.

If it is necessary to change the contents of the INT_MOD or INTVX registers, it has to be made sure that exceptions are denied. In addition, before interrupts are enabled again, it is necessary to execute at least one instruction, for example *nop*. With this it is made sure that the contents of the interrupt registers are updated before new interrupts occur.

Brach instructions can not be executed conditionally. On the contrary register indirect jumps can be conditionally executed. Instruction, which are defined for forming long constants, such as *lui* and *lli* can not be conditionally executed (including *scon*). An instruction following the brach instruction is always executed before transferring to the address specified by the branch. This so called brach slot instruction must be filled with the *nop* instruction, if any other suitable instruction is unawailable. The brach slot **must not** be filled with another branch instruction or with *retu, reti, scall* or *swm* instruction. There **must be** a *nop* instruction after *retu, reti, scall* or *swm* instruction.

Programmming tips

A list of instructions that cannot be conditionally executed:

lui, lli, cmp, cmpi, rcon, scon, retu, reti, bxx (all branch instructions), jmp, jal, swm, trap