**Cycle:**

● if PS(W) = 0 then GOTO STOP; Проверка тумблера работа-останов, стоп если останов

● if PS(IRQ) = 0 then GOTO INFETCH; Если нет прерывания, то на выборку след. команды

● IRQSC ; Сформировать сигнал предоставление прерывания

~0 + SP → SP, AR

● IP → DR ; IP → -(SP)

● DR → MEM(AR

~0 + SP → SP, AR

● PS → DR ; PS → -(SP)

● DR → MEM(AR); а также.

LTOL(CR) → BR ; младшие 8 разрядов CR (номер вектора прерывания) записать в BR

● SHL(BR) → BR, AR ; Вычисляем адрес ячейки с переходом на подпрограмму обработки прерывания, как номер вектора \* 2

● MEM(AR) → DR; адрес обработчика прерывания записать в DR ...

● DR → IP ; … а затем в IP

● LTOL(BR + 1) → AR ; … выбрать адрес следующей ячейки вектора прерывания, ограничивая результат 8-ю разрядами

● MEM(AR) → DR; содержимое PS обработчика прерывания записать в DR ...

● DR → PS ; … а затем установить его в регистр

● if PS (W) = 0 then GOTO STOP; Check

toggle switch work-stop, stop if stop

● if PS (IRQ) = 0 then GOTO INFETCH; If a

there is no interruption, then the fetch trace. commands

● IRQSC; Generate a signal to provide an interrupt

~0 + SP → SP, AR

● IP → DR ; IP → -(SP)

● DR → MEM(AR

~0 + SP → SP, AR

● PS → DR ; PS → -(SP)

● DR → MEM(AR); а также.

LTOL (CR) → BR; lower 8 bits of CR (interrupt vector number) write to BR

● SHL (BR) → BR, AR; We calculate the address of the cell with the transition to the interrupt handling routine, as a vector number \* 2

● MEM (AR) → DR; write the address of the interrupt handler to DR ...

● DR → IP; ... and then to IP

● LTOL (BR + 1) → AR; ... select the address of the next cell of the interrupt vector, limiting the result to 8 bits

● MEM (AR) → DR; write the PS content of the interrupt handler to DR ...

● DR → PS; ... and then set it to register

**IN short :**

1 возникло прерывание

2 проверяем не стоим ли мы

3 есть ли прерывание

4 предоставляем прерывание

5 получаем вектор в регистор команд

6 записываем старые содержимые счетчиков команд состояния стек

7 загружаем их новыми значения из вектора прерывания

1 an interrupt occurred

2 check if we are worth

3 is there an interrupt

4 provide an interrupt

5 we get the vector into the command register

6 write the old contents of the stack state command counters

7 load them with new values from the interrupt vector

**команд нет цикла прерывания?**

SHLT

**Когда будет обработано прерывание, возникшее в момент выполнения обработчика прерывания?**

ну, я думаю, это зависит от того, разрешены ли прерывания во время одного из прерываний. и у каждой команды есть цикл прерывания, поэтому я предполагаю, что это будет сделано после выполнения команды внутри прерывания.

….well, i guess it depends on if interruptions are enabled while one of interruptions is running. and each command has an interruption cycle so i guess it will be done after a running command inside of interruption.

**IRET instruction:**

Инструкция IRET используется в конце процедуры обслуживания прерывания для возврата выполнения прерванной программе. Возврат из обработки прерывания

The IRET instruction is used at the end of an interrupt service procedure to return execution to the interrupted program.

**Приоритет обработки прерываний от нескольких ВУ.**

У бэвм нет программного приоритета у различных прерываний, все прерывания считаются равными друг относительно друга.

Но при этом у нас есть схемный приоритет, т.к. ВУ подключены друг к другу последовательно. Поэтому, если у нас одновременно будут вызваны прерывания с разных ву, то обработается быстрее прерывание ВУ с меньшим номером.

The computer has no software priority for the various interrupts, all interrupts are considered equal relative to each other.

But at the same time, we have circuit priority, since VUs are connected to each other in series. Therefore, if we simultaneously call interrupts from different wu, then the interrupt of the VU with a lower number will be processed faster.

**What is an interrupt vector?**

An interrupt vector is the memory location of an interrupt handler, which prioritizes interrupts and saves them in a queue if more than one interrupt is waiting to be handled.

An interrupt is a signal from a device attached to a computer, or from a program within the computer, that tells the OS (operating system) to stop and decide what to do next. When an interrupt is generated, the OS saves its execution state by means of a context switch, a procedure that a computer processor follows to change from one task to another while ensuring that the tasks do not conflict. Once the OS has saved the execution state, it starts to execute the interrupt handler at the interrupt vector.