**ВОПРОС 22**

Организация прерываний на примере i8086

Существуют различные методы поиска источника запроса на прерывание. Как мы помним у процессора 8086 имеется два входа запросов на прерывание: NMI и INTR. Вот когда приходит запрос на прерывание, то процессору каким-то образом требуется узнать какое конкретное устройство отправило данный запрос. Существует два распространённых способа:  
1) (Примитивно) Это устройство перед тем как подать запрос на прерывание пишет определённый бит в определённый регистр и процессор, когда ему требуется определить источник запроса на прерывание, он обращается к этому регистру и считывает содержимое регистра и в результате узнает какое устройство отправило данный запрос

2) (Наиболее распространённый) векторное прерывание. Когда используется контроллер прерываний, который передаёт номер прерывания процессору и процессор таким образом уже знает, какое устройство конкретно сформировало данный запрос. Такой способ - векторное прерывание. Вектором (в системе прерываний) называется адрес обработчика прерывания, который хранится в специальной таблице векторов прерываний. Когда контроллер передаёт номер прерывания процессору, процессор по этому номеру извлекает адрес обработчика этого прерывания.

Адрес в процессоре 8086 20-разрядный и формируется с помощью содержимого двух регистров, а именно с помощью сегментов кода CS и с помощью регистра счётчика команд IP. Так как эти два регистра 16-разрядные, то есть на хранение содержимого каждого регистров требуется нам два байта, то у нас каждый вектор занимает 4 байта. И вот эти все вектора, то есть все адреса обработчиков прерываний хранятся в начальном сегменте памяти и занимает 1024 байта, всё это называется таблицей векторов прерываний.

**Таблица векторов прерываний**

В реальном режиме работы в системе прерываний используется понятие вектора прерывания.

Каждый вектор прерывания состоит из четырех байтов, или двух слов: первые два содержат новое значение для регистра IP, а следующие два — новое значение для регистра CS. Таблица векторов прерывания занимает 1024 байт. Таким образом, в ней может быть задано 256 векторов прерываний. В процессоре i8086 эта таблица располагается на адресах 00000H-003FFH.

Вся эта таблица располагается в начальном участке памяти.

**Таблица векторов прерываний**

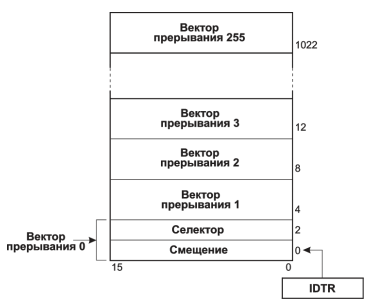
Нулевой вектор прерывания состоит из селектора и смещения.

В качестве селектора выступает содержимое сегментного регистра CS

В качестве смещения содержимое регистра счётчика IP

Каждое содержимое регистра требует 2 байта, соответственно каждый вектор занимает 4 байта памяти.

Если знаем номер вектора и знаем, что вектора последовательно расположены и каждый занимает 4 байта. Значит, можно номер умножить на 4, чтобы найти физический адрес прерывания.



**Виды прерываний**

Прерывания бывают двух видов: внутренние и внешние.

Внутренние прерывания возникают в результате работы процессора в ситуациях, которые нуждаются в специальном обслуживании, или при выполнении специальных команд (INT, INTO). (иначе говоря, ситуация внутри процессора)

Это следующие прерывания:

- прерывание при делении на ноль (номер прерывания 0);

- прерывание по флагу TF (номер прерывания 1); - трассировка (для отладки)

- прерывания, возникающие при выполнении команд INT называются программными.

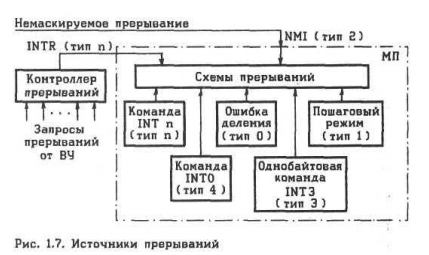
В качестве операнда команды INT указывается номер прерывания, которое нужно выполнить.

Внешние прерывания возникают по сигналу какого-нибудь внешнего устройства. Существует два специальных внешних сигнала среди входных сигналов процессора, при помощи которых можно прервать выполнение текущей программы и тем самым переключить работу центрального процессора. (иначе говоря, ситуации внешними по отношению к процессору устройству)

NMI (No Mask Interrupt — немаскируемое прерывание) - прерывание будет обработано в любом случае

INTR (Interrupt Request — запрос на прерывание) - обработка в случае, если флаг даёт это сделать. Иначе игнорирует.

**(то что внутри контура это всё процессор и соответственно внутренние прерывания)**

****

**Этапы обработки прерываний**

1. Контроллер прерываний получает заявку от определенного периферийного устройства и, соблюдая схему приоритетов, генерирует сигнал INTR (запрос на прерывание), который является входным для микропроцессора.

2. Микропроцессор проверяет флаг IF в регистре флагов. Если он установлен в 1, то переходим к шагу 3. В противном случае работа процессора не прерывается.

3. Микропроцессор генерирует сигнал INTA (подтверждение прерывания). В ответ на этот сигнал контроллер прерываний посылает по шине данных номер прерывания.

4. В стек помещается регистр флагов. (регистр флагов может меняться, потому что нужно будет вернуться в программу без изменений) - 2 байта

5. Флаг включения-выключения прерываний IF и флаг трассировки TF, находящиеся в регистре флагов, обнуляются для блокировки других маскируемых прерываний и исключения пошагового режима исполнения команд. (вначале отключается, остальное разработчик обработки прерываний решает)

6. Значения регистров CS и IP сохраняются в стеке. (чтобы вернуться к прерванной точке) - 2 байта и 2 байта

7. Вычисляется адрес вектора прерывания и из вектора, соответствующего номеру прерывания, загружаются новые значения IP и CS.

8. Выполнение подпрограммы обработчика прерывания. (менять содержимое регистров флагов можем, к примеру)

9. Извлечение из стека IP и CS (восстановление точки в программе)

10. Извлечение из стека регистра флагов.

11. Процессор продолжает работу с того места, где он был прерван

**Использование прерываний**

- для обмена информацией между процессором и внешним устройством (ВУ); (позволяет обмениваться информацией с устройствами разных скоростей без замедления процессора)

- в аварийных ситуациях, например, при понижении напряжения питания; (для сохранения данных, к примеру)

- при исключительных условиях, таких, как переполнение, (деление на ноль);

- для индикации аппаратных сбоев, приводящих к ошибкам при обработке данных; (ошибка контроля чётности, переспросить искажённый пакет)

- при программных сбоях;

- для координации работы в многопроцессорных системах; (пересылка данных, к примеру)

- для профилактики, ремонта, тестирования и отладки системы (TF) (просмотр содержимого регистра и всё такое)