КАРХ: Тема_15: Архитектура Intel®64

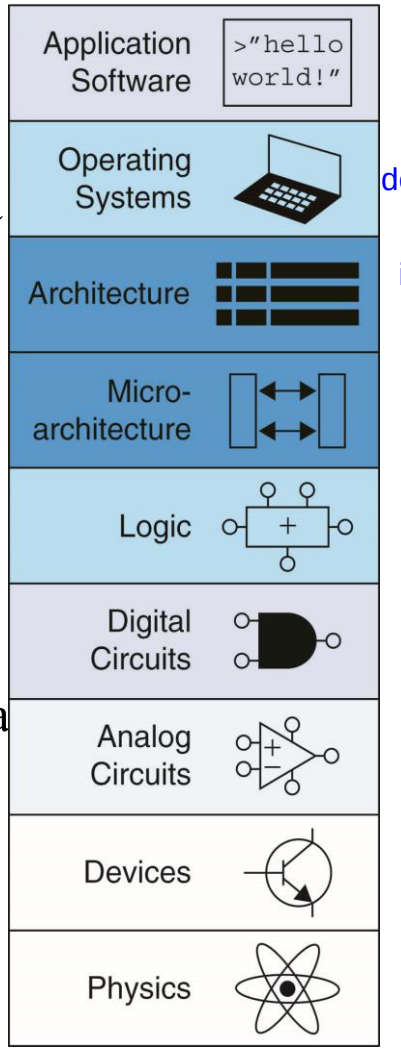
Intel®64 и IA-32 архитектури.

- IA-32 архитектурата поддържа 3 основни работни мода:

– защитен мод (**protected mode**). Този мод отговаря на естественото състояние на процесора при работа в многозадачна среда (multi-tasking environment);

– **real-address mode** – това е програмната среда на стария **Intel 8086** процесор (заедно с разширенията). В този мод процесорът влиза след стартиране (power-up) или ресетиране (Reset).

– **system management mode (SMM).** Този мод дава на операционната система (ОС) или друга служебна програма прозрачен механизъм за въвеждане на специфични функции като *power management* и *system security*. Процесорът влиза в **SMM** след активиране на *external SMM interrupt pin (SMI#)* или заявка за прекъсване от *Advanced Programmable Interrupt Controller* (APIC).



КАРХ: Тема_15: Архитектура Intel®64

Intel®64 архитектура.

- Intel®64 архитектурата добавя нов мод – **IA-32e mode**, който съдържа два под-мода (sub-modes):

– **Съвместим мод (Compatibility mode)** – позволява на повечето наследени 16 bit и 32 bit приложения (програми) да вървят без пре-компилация под 64 bit операционна система. Този мод е подобен на 32 bit защитен мод. Приложенията имат достъп само до първите 4GB от линейното адресно пространство. Използват се 16 bit и 32 bit адреси и операнди. Допуска се използване и на адресно пространство над 4GB чрез използване на PAE (Physical Address Extensions).

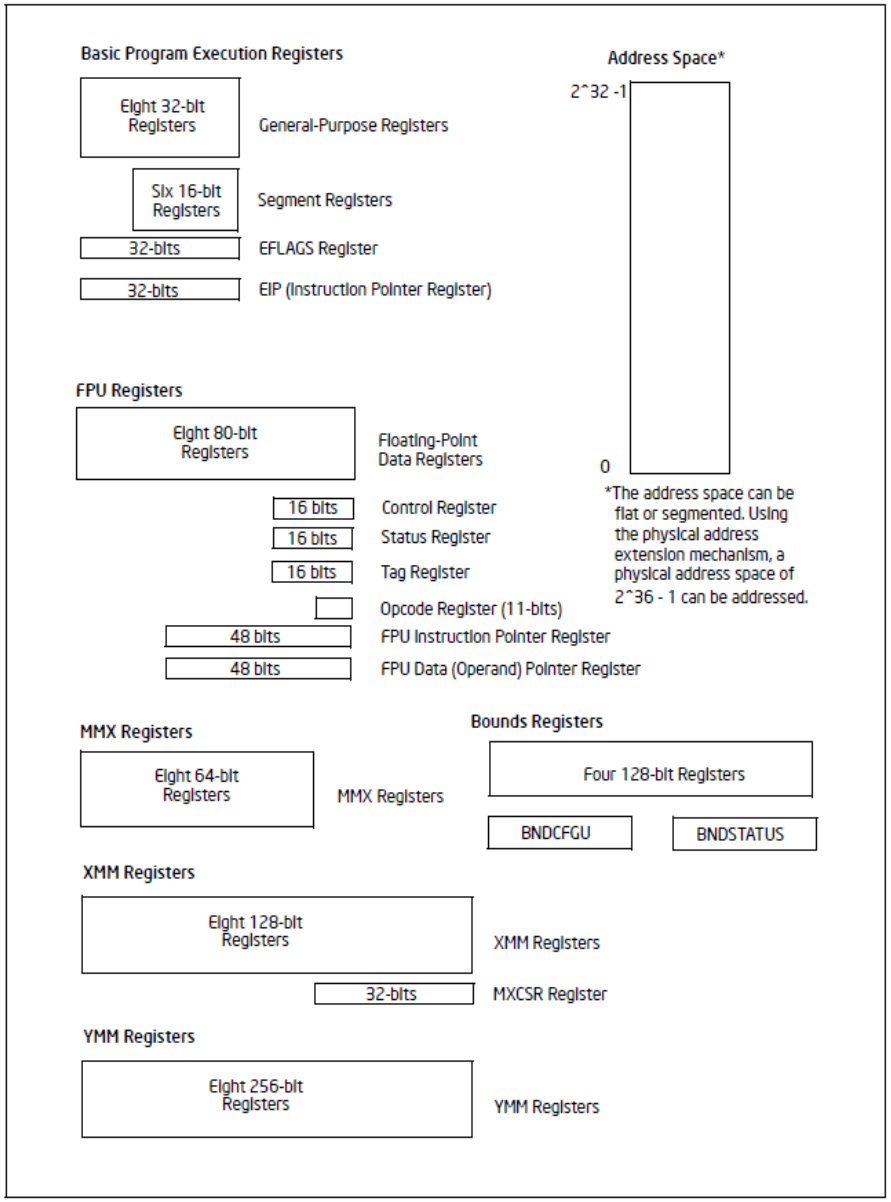
– **64 bit mode** – този мод позволява на 64 bit операционна система да стартира приложения използващи 64 bit адресно пространство. Този подмод може да се третира като 64 bit мод на IA-32 архитектура. Освен това се увеличава броят на регистрите с общо предназначение и SIMD регистрите от 8 на 16, като първите се удължават до 64 bit. Въвежда се нов opcode prefix (REX), осигуряващ на командите достъп до удължените части на регистрите. Този мод се разрешава от операционната система на код-сегментна основа. По подразбиране адресите са 64 bit , а операндите – 32 bit. Размерът на операндите може да се увеличи с помощта на инструкции използващи REX префикс. Това позволява на множество инструкции да използват 64 bit регистри и 64 bit адресиране.

КАРХ: Тема_15: Архитектура Intel®64

Основна работна среда

(basic execution environment)

- **Адресно пространство** – всяка програма може да адресира линейно адресно пространство до 4 GB и физичсско адресно пространство до 64 GB.
- **Основни програмно достъпни регистри** – 8 регистри с общо предназначение, 6 сегментни регистри, EFLAGS регистър, EIP (instruction pointer) регистър. Използват се в общи инструкци, работещи с целочислена математика върху байт, дума и двойна дума константи.
- **x87 FPU регистри** – 8 x87 FPU регистри за данни, x87 FPU control register, status register, x87 FPU instruction pointer register, x87 FPU operand (data) pointer register, x87 FPU tag register, x87 FPU opcode register за работа с числа с плаваща точка, с единична и двойна точност и с двоично-кодирани десетични числа.



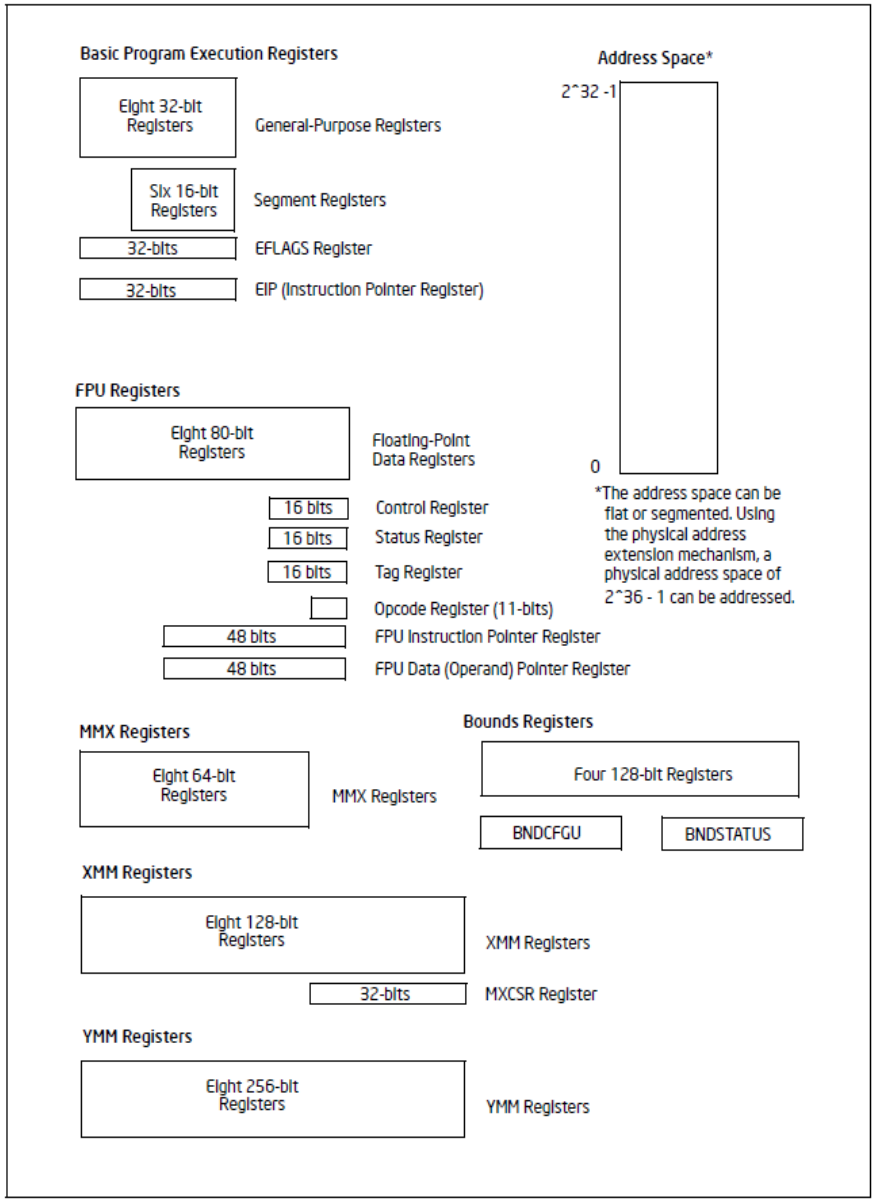
IA-32 Basic Execution Environment for Non-64-bit Modes

КАРХ: Тема_15: Архитектура Intel®64

Основна работна среда

(basic execution environment)

- **MMX регистри** – 8 MMX регистри за поддръжка на SIMD (single-instruction multiple-data) за работа с 64 bit пакетирани байтове, думи и двойни думи от целочислени константи.
- **XMM регистри** – 8 XMM регистри за данни и MXCSR регистър за поддръжка на SIMD операции върху 128 bit пакетирани байтове, думи, двойни думи и четворни думи от целочислени константи, числа с единична и двойна точност с плаваща точка, пакетирани в 128 bit.
- **YMM регистри** – същото като XMM регистрите, но в 256 bit.
- **Bounds registers (регистри на границите)** - BND0 ÷BND3 съхраняват долната и горната граници (64 bit всеки) на memory buffer при изпълнение на MPX инструкции.



IA-32 Basic Execution Environment for Non-64-bit Modes

КАРХ: Тема_15: Архитектура Intel®64

Основна работна среда (basic execution environment)

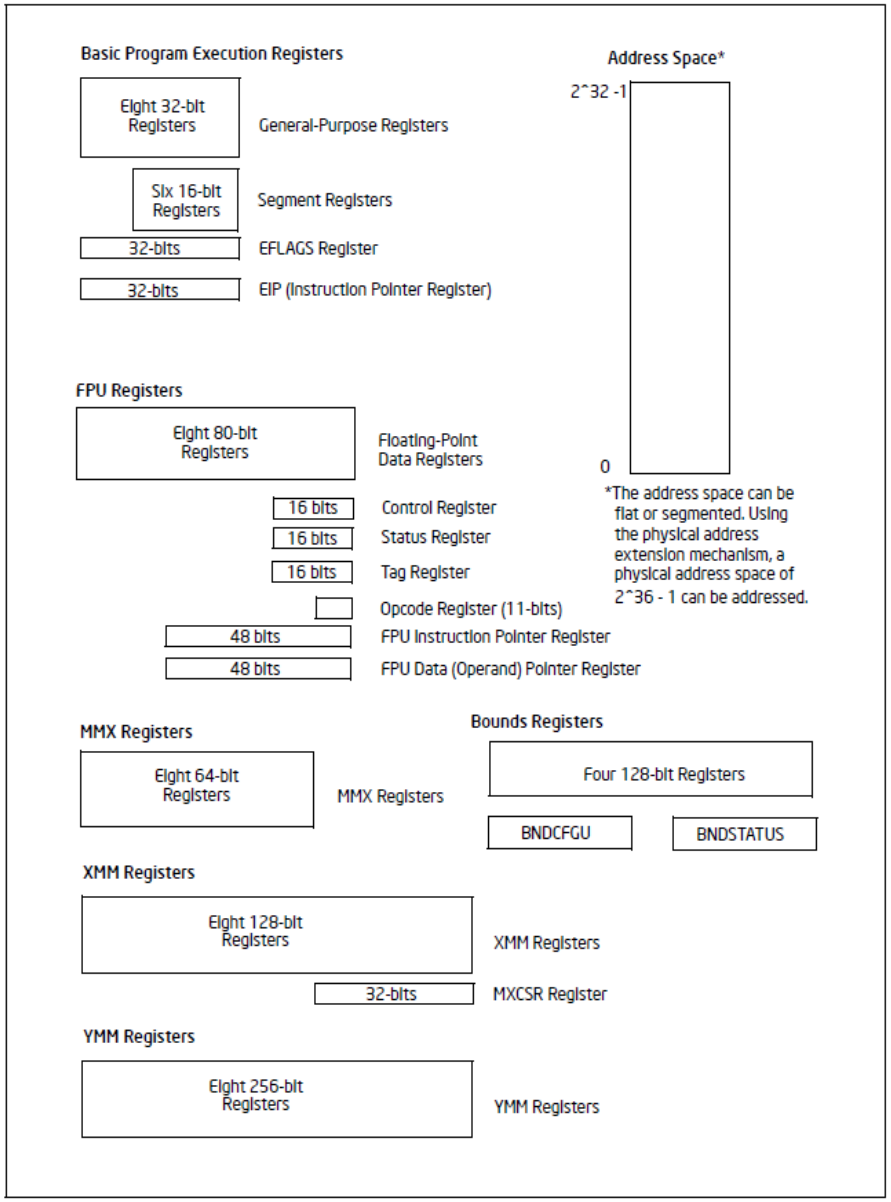
- Стек – локализиран в паметта. Необходим при процедури и подпрограми за запазване на параметри и стойности на регистри.

- Входно-изходни портове (I/O ports) – за трансфер на данни от/в системата.

- Контролни регистри (Control registers) – 5 контролни регистри (CR0÷CR4) – определят работния мод на процесора и характеристиките на текущо изпълняваната задача.

- Memory Management Registers – GDTR, IDTR , task register, LDTR – показват местоположението на структурите от данни в защитен мод на работа на паметта.

- Debug registers – DR0÷DR7–за наблюдение на процесорните debugging operations.

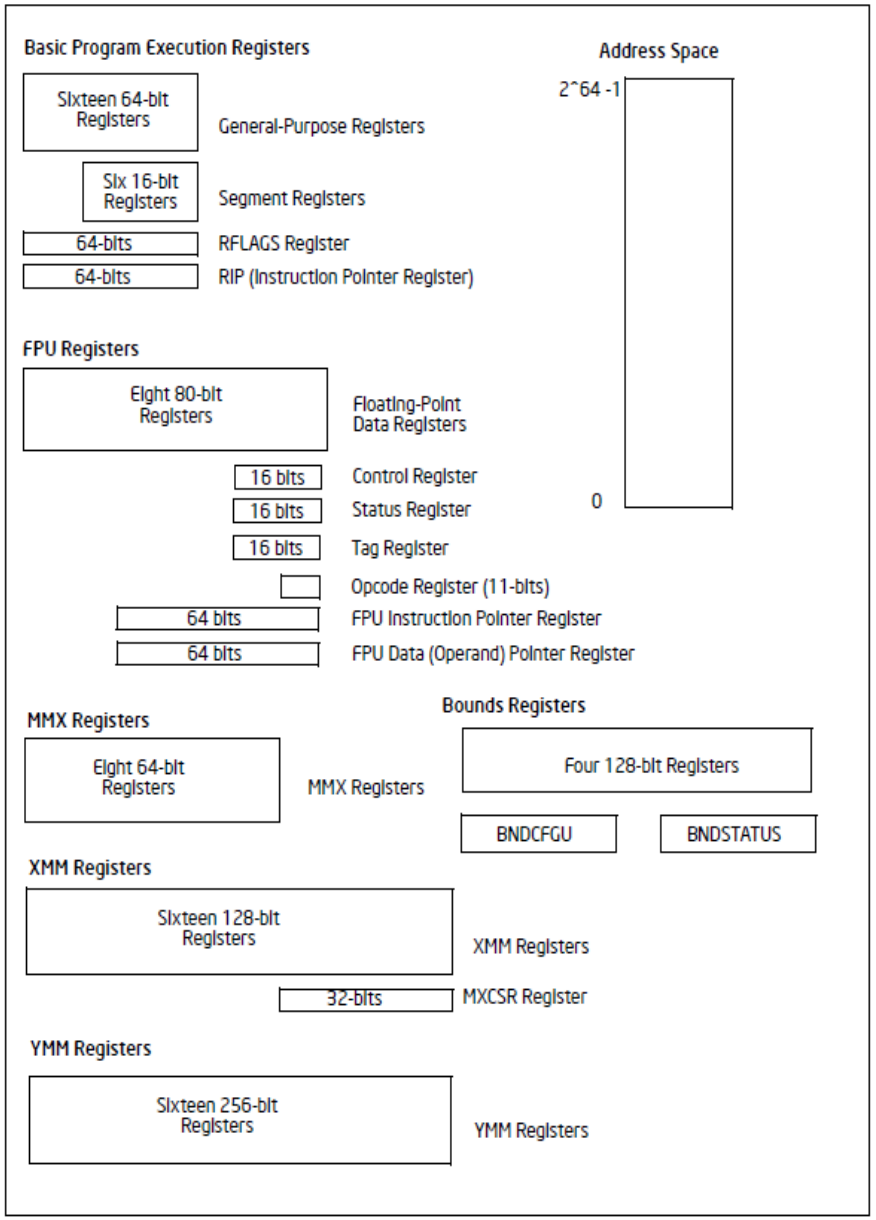


IA-32 Basic Execution Environment for Non-64-bit Modes

КАРХ: Тема_15: Архитектура Intel®64

Работна среда в 64 bit мод (64bit mode execution environment)

- **Адресно пространство** – задачи или програми, работещи в този мод могат да адресират линейно адресно пространство от 2^{64} байта и физическо адресно пространство до 2^{46} байта.
- **Основни програмно достъпни регистри** – 16 регистри (64 bit) с общо предназначение, 6 сегментни регистри, RFLAGS (64 bit) регистър, старшите 32 bit – запазени, а младшите 32 bit остават като EFLAGS, EIP (instruction pointer) регистър става 64 bit . Използват се в общи инструкции, работещи с целочислена математика върху байт, дума, двойна дума и четворна дума константи.
- **XMM регистри** –16 XMM регистри за SIMD

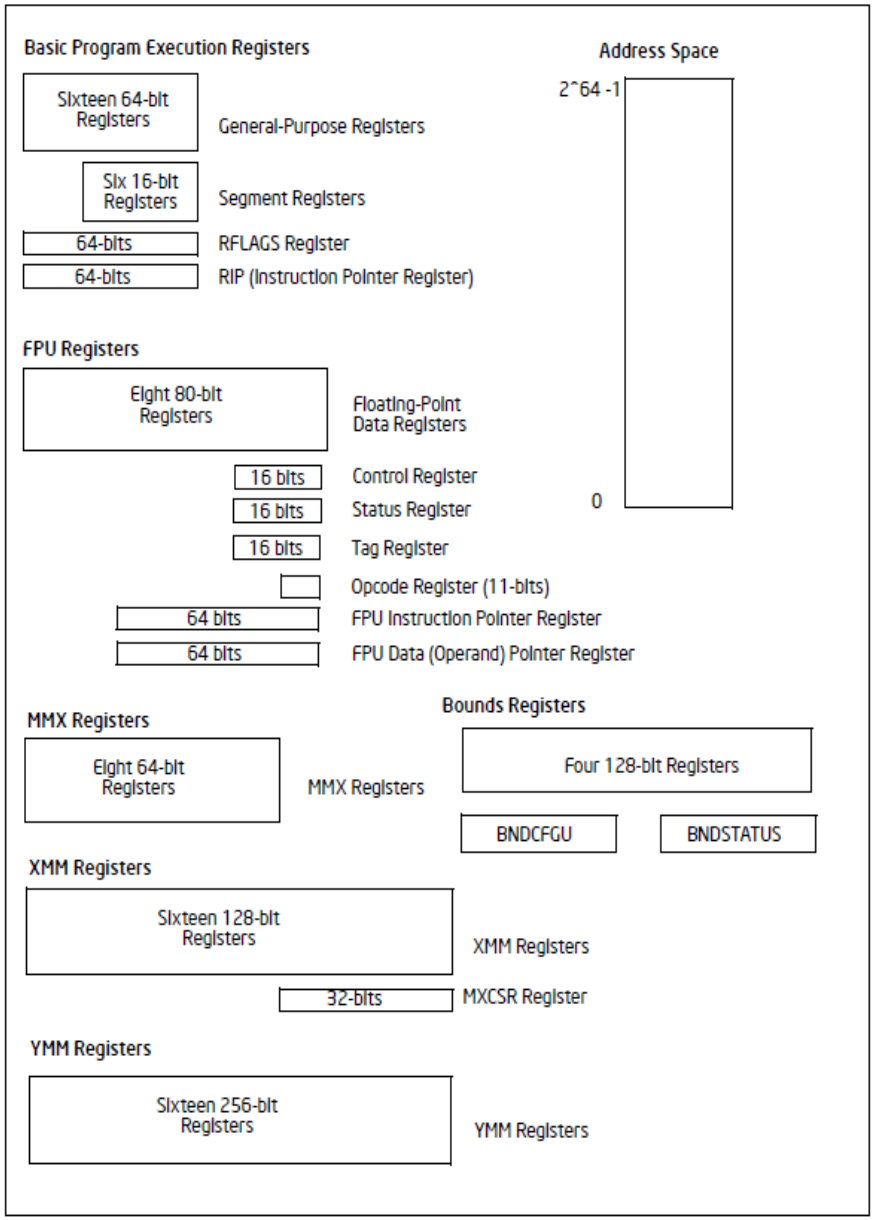


64-Bit Mode Execution Environment

КАРХ: Тема_15: Архитектура Intel®64

Работна среда в 64 bit мод (64bit mode execution environment)

- **YMM регистри** –16 YMM регистри за SIMD
- **Стек** – стек поинтъра става 64 bit
- **Контролни регистри (Control registers)** – удължават се до 64 bit. Добавя се нов регистър TPR (task priority register).
- **Debug registers** – удължават се до 64 bit.
- **Descriptor table registers** – global descriptor table register (GDTR), interrupt descriptor table register (IDTR) се удължават до 80 bit, a local descriptor table register (LDTR), task register (TR) – до 64 bit, така че да могат да съдържат пълен 64 bit базов адрес.



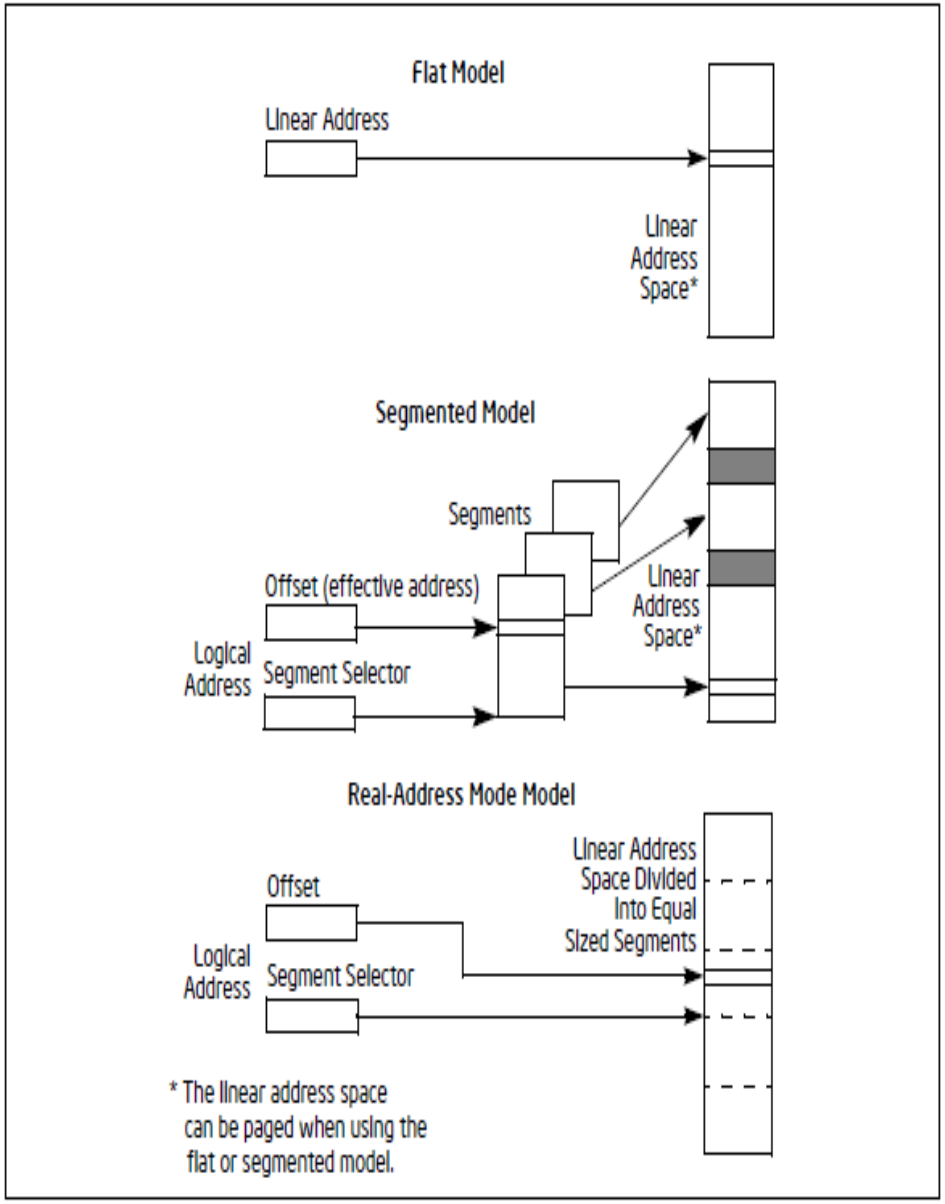
64-Bit Mode Execution Environment

КАРХ: Тема_15: Архитектура Intel®64

Организация на паметта

(IA-32 Memory Models)

- **Плосък модел (Flat Memory Model)** – за програмите паметта изглежда като едно непрекъснато адресно пространство – *линейно адресно пространство* – програмният код, данните и стекът са разположени в това пространство.
- **Сегментен модел (Segmented Memory Model)** - за програмите паметта изглежда като разделена на групи от адресни пространства наречени *сегменти*. Програмният код, данните и стекът са разположени в отделни сегменти. За адресиране на байт от даден сегмент се използва *логически адрес,* който съдържа сегментен селектор и офсет. Първият определя сегмента, който трябва да се избере, а вторият – отместването в адресното пространство на сегмента.

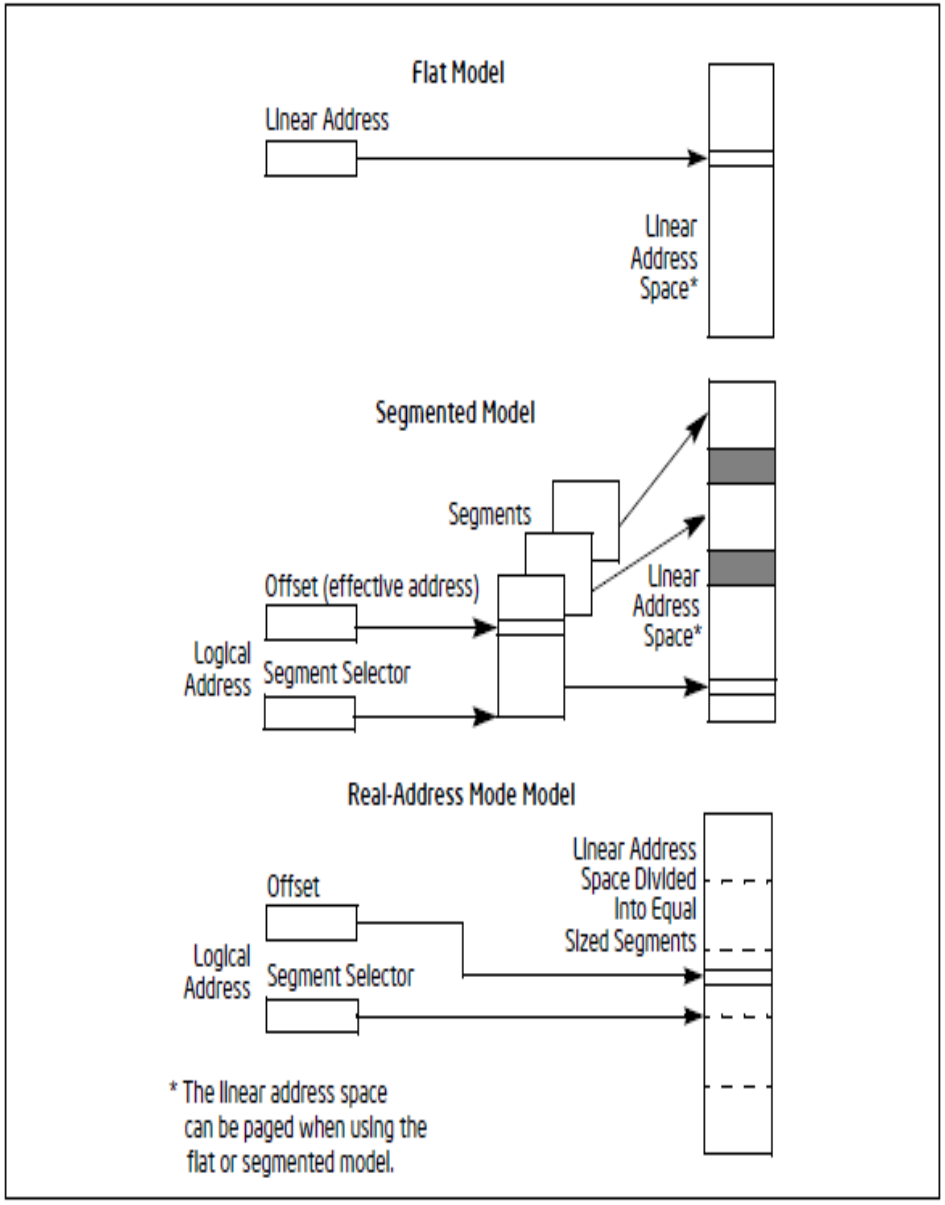


КАРХ: Тема_15: Архитектура Intel®64

Организация на паметта (IA-32 Memory Models)

Програмите могат да адресират до 16,383 сегмента от различен тип и размер (до 2^{32} байта всеки). Реално всички сегменти се намират в линейното адресно пространство. За това логическите адреси се транслират в линейни адреси от процесора.

- **Модел с реални адреси (Real-address Mode Memory Model)** – това е моделът на паметта на Intel 8086 процесор, който е запазен с цел съвместимост със старите програми. Паметта е разделена на сегменти с фиксиран размер от 64KB за програмите и операционната система. Максималният размер на линейното адресно пространство при този модел е 2^{20} байта.



КАРХ: Тема_15: Архитектура Intel®64

Връзка работен мод – модел на паметта.

- В **защитен мод** процесорът може да използва всеки от описаните модели на паметта, като изборът на модел зависи от операционната система или изпълнявания код. В режим на многозадачност отделните приложения могат да използват различни модели на паметта.
- В **real-address mode** може да се използва само real-address mode memory model.
- При **system management mode (SMM)** използваният модел на паметта е близък до този на real-address mode memory model.
- Софтуерът, който използва **съвместим мод,** използва същия модел на паметта, както ако би използвал 32 bit защитен мод.
- В **64-bit mode** сегментирането е изобщо (но не напълно) изключено. Създава се плоско 64-bit линейно адресно пространство. Процесорът разглежда сегментните регистри CS, DS, ES и SS като нулеви (с нулева стойност) в 64-bit мод. Сегментният модел и моделът с реални адреси на паметта са неприложими при 64-bit мод.

КАРХ: Тема_15: Архитектура Intel®64

Основни програмно достъпни регистри.

- В IA-32 архитектурата има 16 основни програмно достъпни регистри, които могат да се групират по следния начин:

– Регистри с общо предназаначение – осем 32 bit регистри – EAX, EBX, ECX, EDX, ESI, EDI, EBP и ESP, които се използват като операнди при логически и аритметични операции, при изчисления на адреси и достъп до паметта.

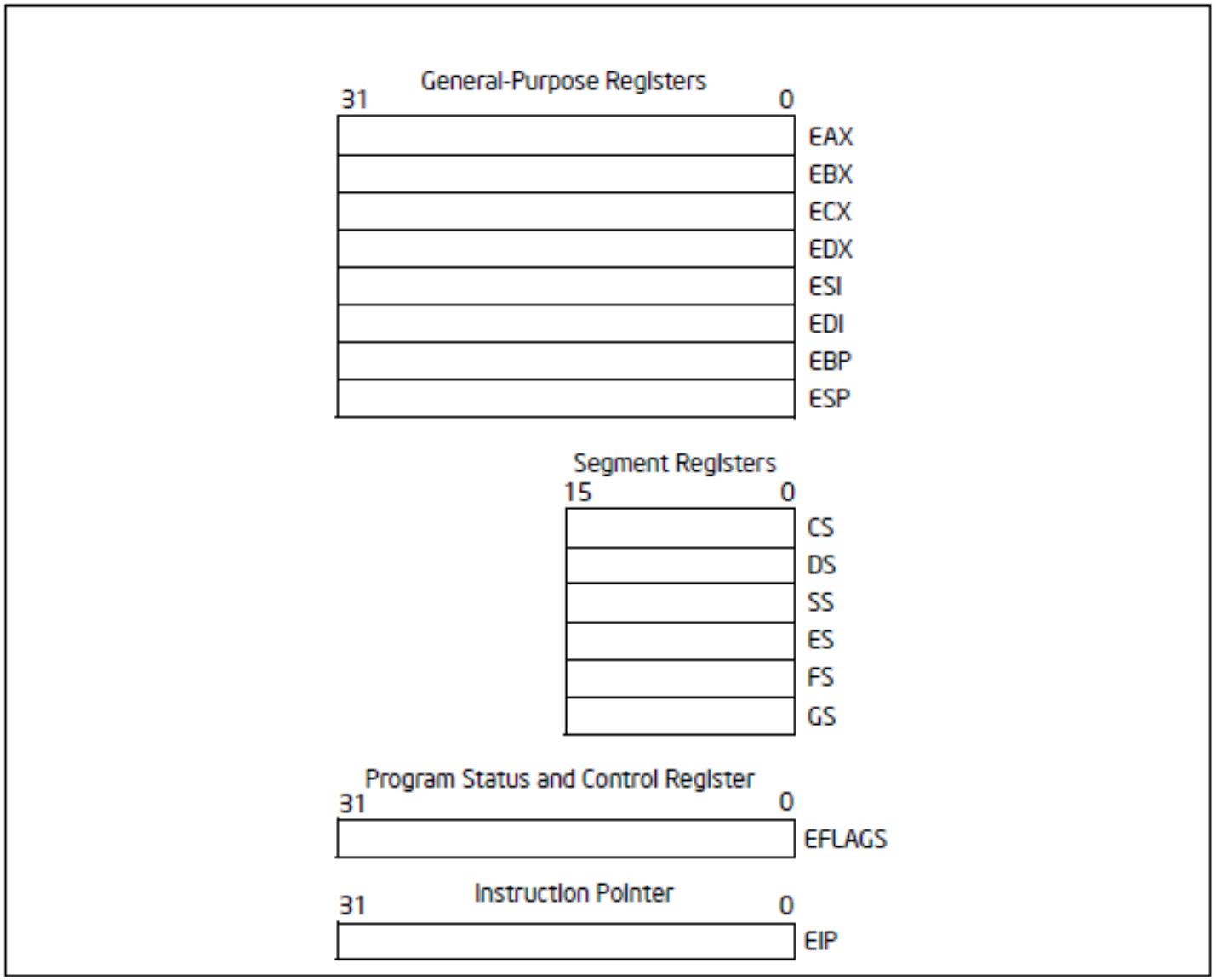
– Сегментни регистри – шест 16 bit регистри – CS, DS, SS, ES, FS и GS, които се използват като сегментни селектори.

– EFLAGS (program status and control) register – съдържа статуса на изпълняваната програма и позволява ограничен (на програмно ниво) контрол над процесора.

– EIP (instruction pointer) register – съдържа 32 bit показалец (pointer) към следващата инструкция, която трябва да се изпълни.

КАРХ: Тема_15: Архитектура Intel®64

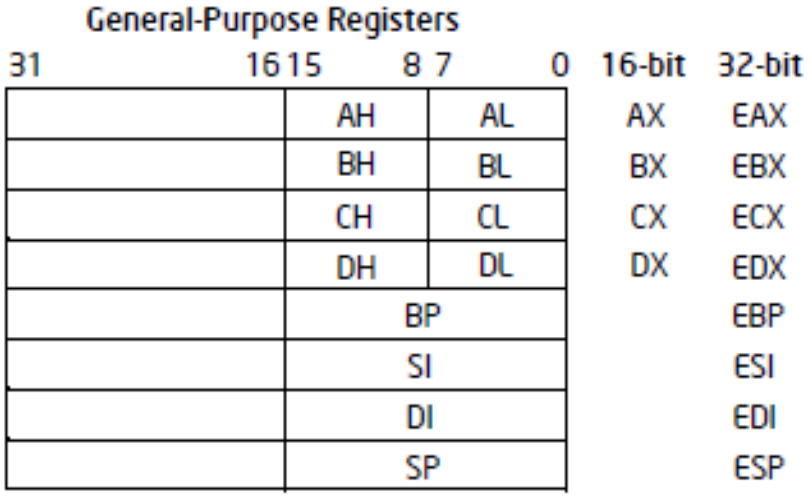
Основни програмно достъпни регистри.



КАРХ: Тема_15: Архитектура Intel®64

Използване на основните програмно достъпни регистри.

- **EAX** – акумулатор за операнди и резултати
- **EBX** – показалец за данни в DS (data segment)
- **ECX** – брояч за стрингове и цикли
- **EDX** – I/O показалец (pointer)
- **ESI** – показалец за данни в сегмент, посочен чрез DS регистъра; показалец за източник при стрингови операции
- **EDI** – показалец за данни (или посока) в сегмент, посочен чрез ES регистъра; показалец за крайна цел при стрингови операции
- **ESP** – показалец на стека (в SS сегмент)
- **EBP** – показалец за данни в стека (в SS сегмент)
- Младшите 16 bit на регистрите EAX, EBX, ECX, EDX, ESI, EDI, EBP и ESP могат да се използват самостоятелно за съвместимост с 16 bit микропроцесори 8086 и 80286.
- Всеки от низшите два байта на регистрите EAX, EBX, ECX, EDX също могат да се използват самостоятелно, напр. AH, BH, AL, BL и т.н.



КАРХ: Тема_15: Архитектура Intel®64

Основни програмно достъпни регистри в 64 bit мод.

- В 64 bit мод има 16 основни програмно достъпни регистри, а размерът на операндите по подразбиране е 32 bit, като регистрите могат да работят както с 32 bit, така и с 64 bit операнди.
- За генериране на 64 bit операнди се използва REX префикс или регистрите R8÷R15
- Всеки от тези регистри е достъпен на ниво байт, дума, двойна дума и четворна дума

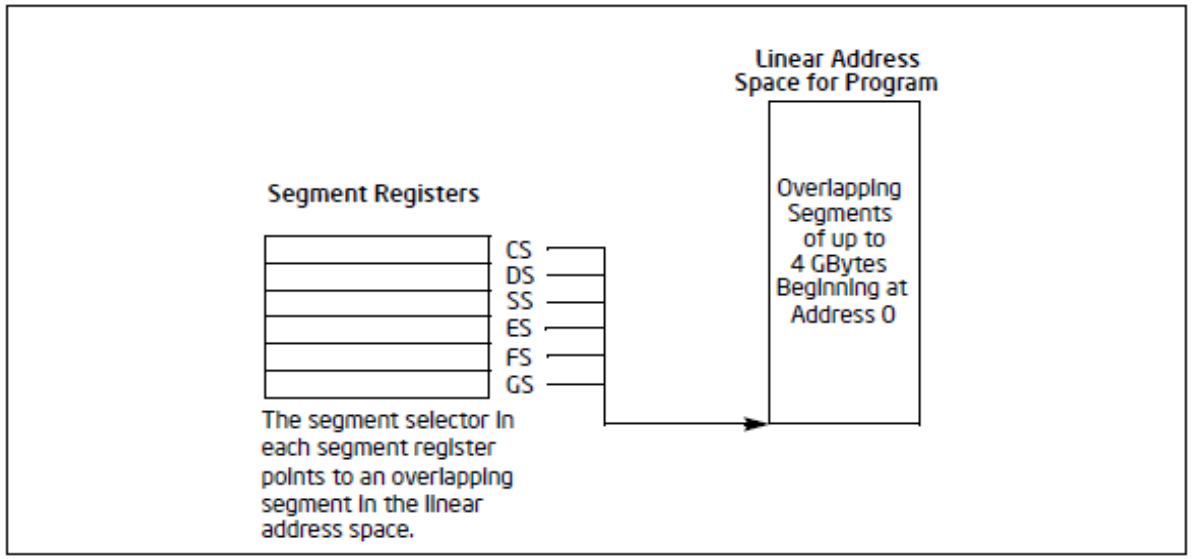
Register Type	Without REX	With REX
Byte Registers	AL, BL, CL, DL, AH, BH, CH, DH	AL, BL, CL, DL, DIL, SIL, BPL, SPL, R8L - R15L
Word Registers	AX, BX, CX, DX, DI, SI, BP, SP	AX, BX, CX, DX, DI, SI, BP, SP, R8W - R15W
Doubleword Registers	EAX, EBX, ECX, EDX, EDI, ESI, EBP, ESP	EAX, EBX, ECX, EDX, EDI, ESI, EBP, ESP, R8D - R15D
Quadword Registers	N.A.	RAX, RBX, RCX, RDX, RDI, RSI, RBP, RSP, R8 - R15

- Има известни ограничения при използване на регистрите като операнди, когато те са с различен размер, в една инструкция.
- В 64 bit мод:
  - 64 bit операнди генерират 64 bit резултат в 64 bit регистър крайна цел;
  - 32 bit операнди генерират 32 bit резултат в 64 bit регистър крайна цел (zero-extended)
  - 8 bit и 16 bit операнди генерират 8 bit и 16 bit резултат в 64 bit регистър крайна цел, като горните 56 (48) bit на регистъра не се променят.
  - Старшите 32 bit на всички 64 bit регистри са *незапазени* при преход в 32 bit мод.

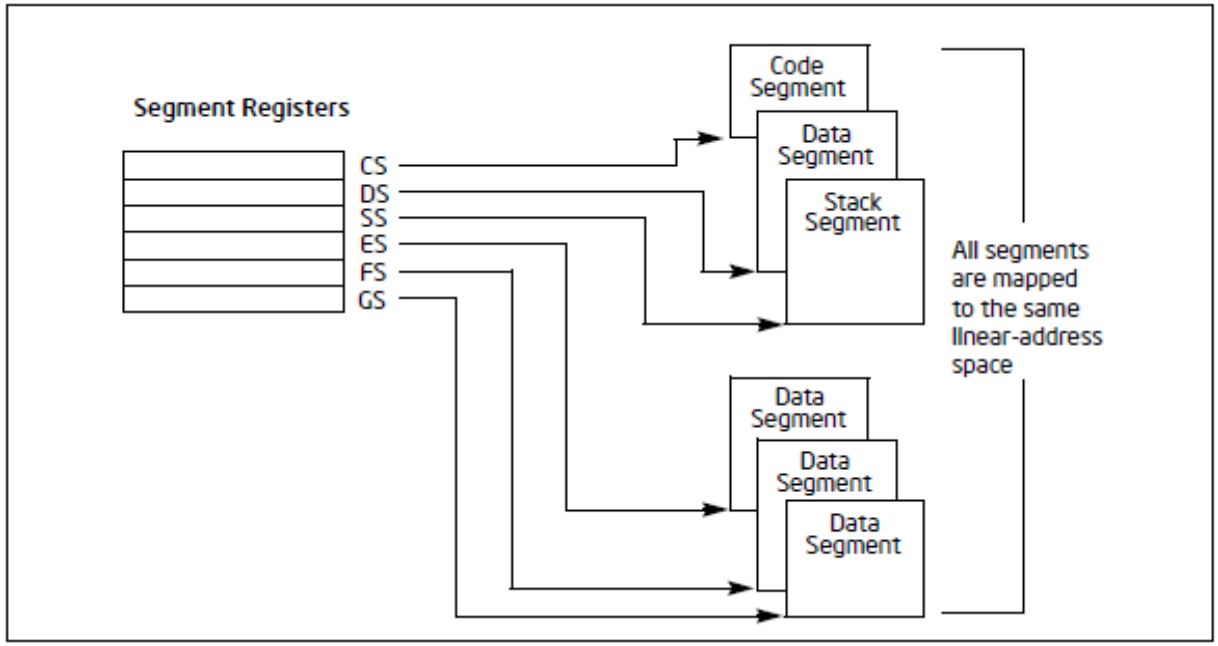
КАРХ: Тема_15: Архитектура Intel®64

Сегментни регистри (Segment Registers)

- **Сегментните регистри** – (CS, DS, SS, ES, FS и GS) съдържат 16 bit сегментни селектори. Това са специални показалци указващи сегмент в паметта. Работата им зависи от избрания от операционната система модел на паметта.
- При **плосък модел** селекторите указват сегменти, които се припокриват и започват от Адрес 0 на линейното адресно пространство.
- При **сегментен модел** на паметта селекторите указват различни сегменти в линейното адресно пространство.



Use of Segment Registers for Flat Memory Model



Use of Segment Registers in Segmented Memory Model

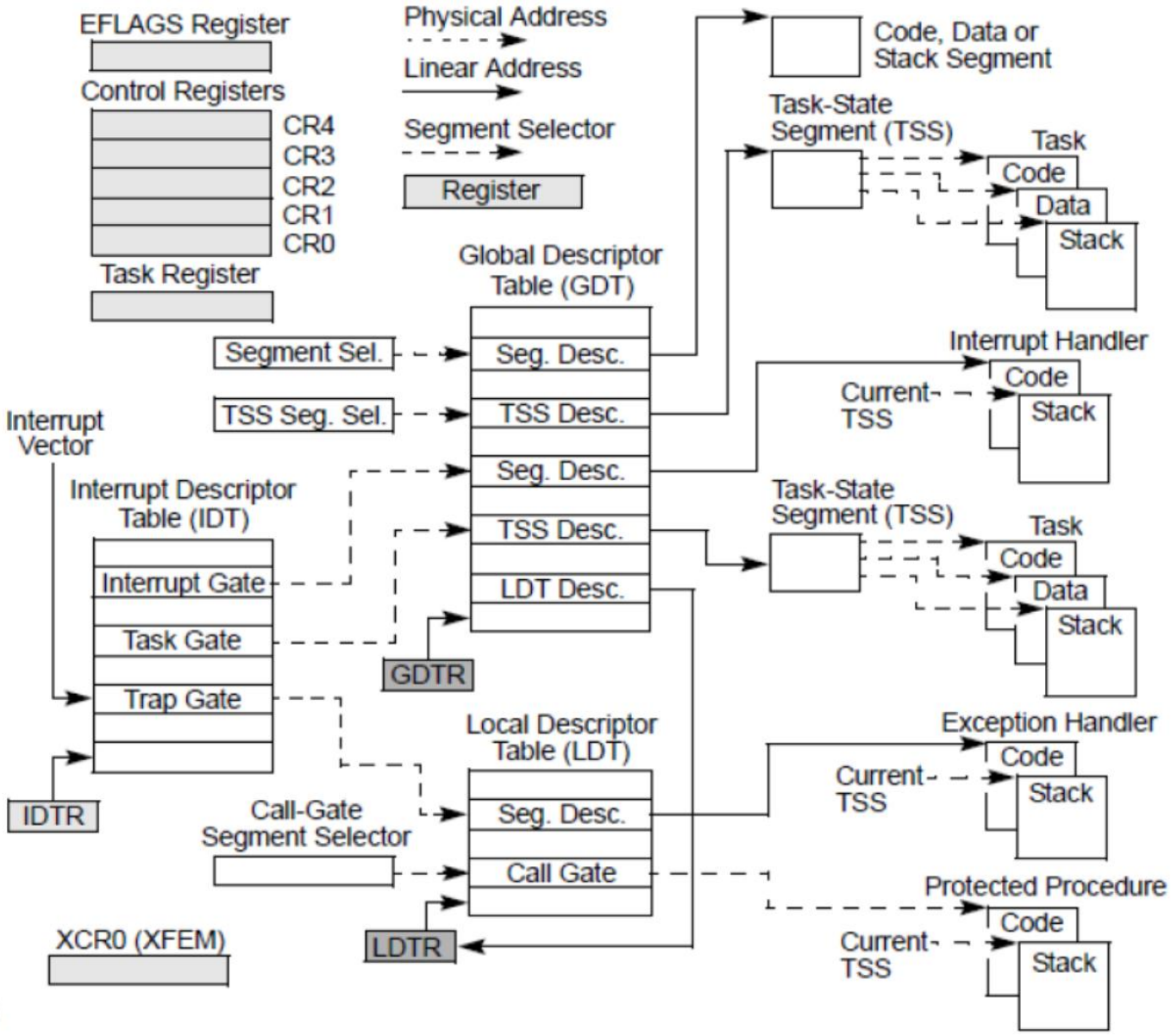
КАРХ: Тема_15: Архитектура Intel®64

Приложение на сегментните регистри.

- Всеки сегментен регистър се асоцира с това, какъв тип данни се съхраняват в даден сегмент – програмен код, данни или стек, към който сочи неговия селектор.
- В **CS** регистъра се съдържа сегментният селектор на сегмента за програмен код (**code segment**). Процесорът извлича следващата инструкция от този сегмент като използва логически адрес (базов адрес CS + офсет EIP).
- **DS, ES, FS** и **GS** регистрите сочат към сегменти за данни (**data segments**), от различен тип, напр. данни на даден модул, данни на модул от по-високо ниво, динамично създадени данни, данни споделени с други програми.
- **SS** регистъра съдържа сегментния селектор на стека (**stack segment**). Всички стекови операции използват този регистър.
- Регистрите CS, DS, SS, ES са въведени при процесорите 8086 и 80286. FS и GS са добавени при 80386 при формирането на архитектурата IA-32.

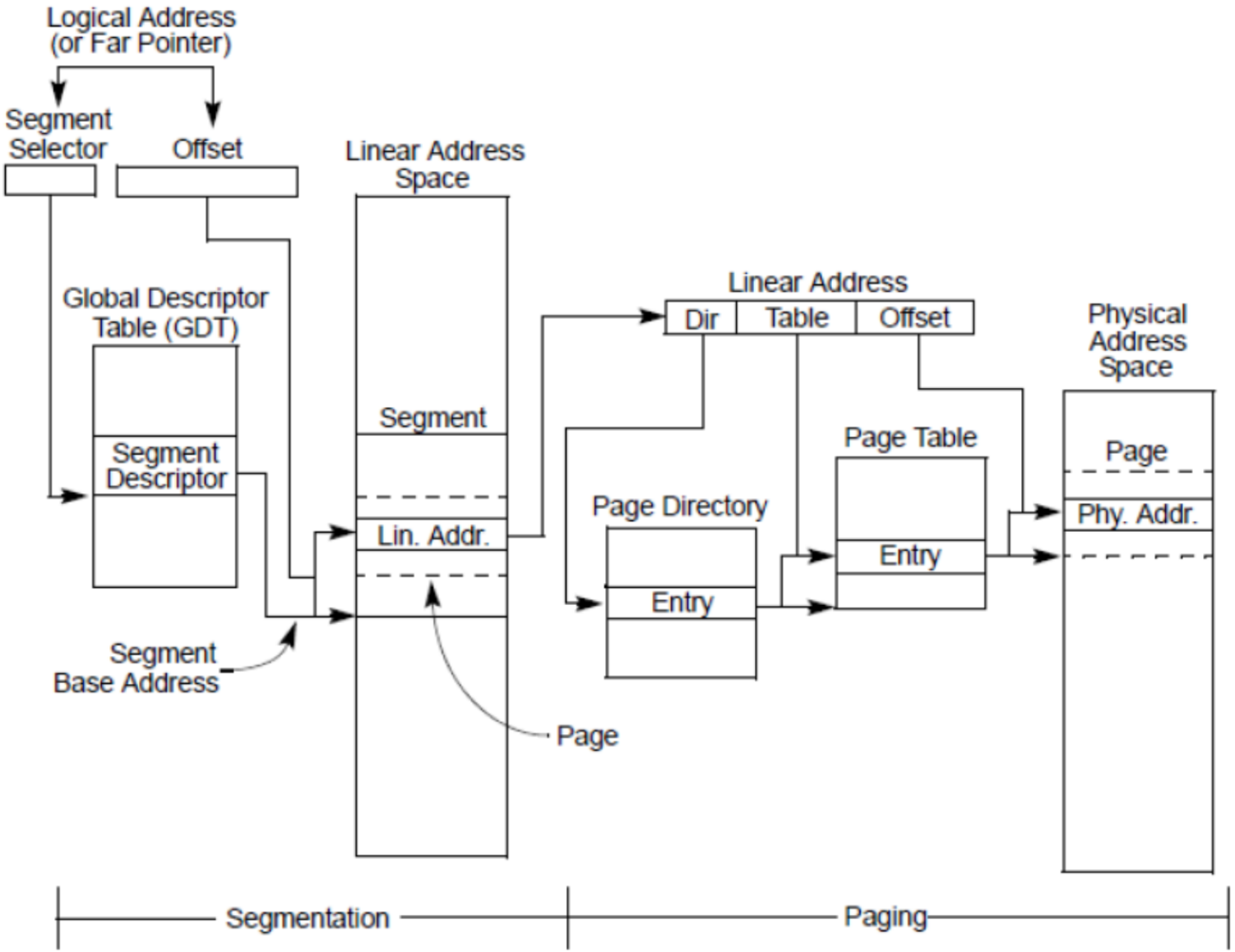
КАРХ: Тема_15: Архитектура Intel®64

Сегментна и странична преадресация.



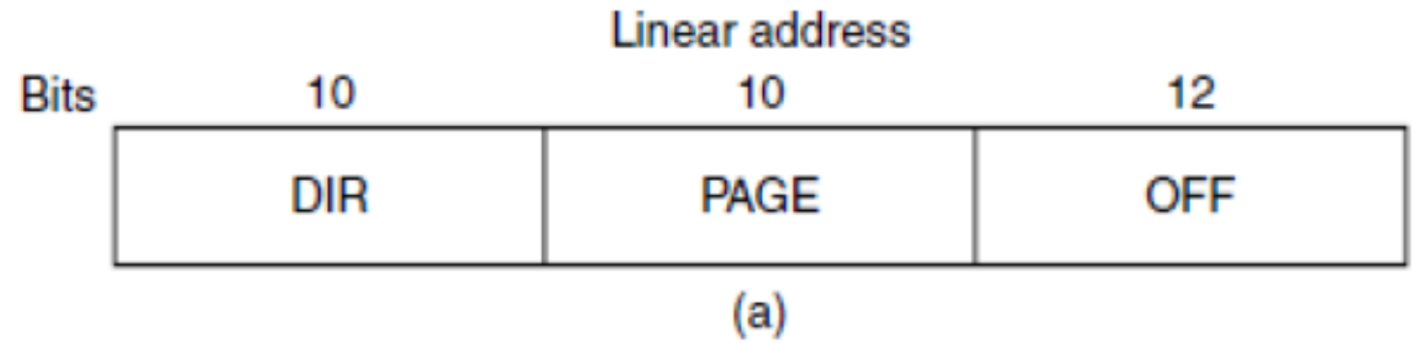
КАРХ: Тема_15: Архитектура Intel®64

Сегментна и странична преадресация.

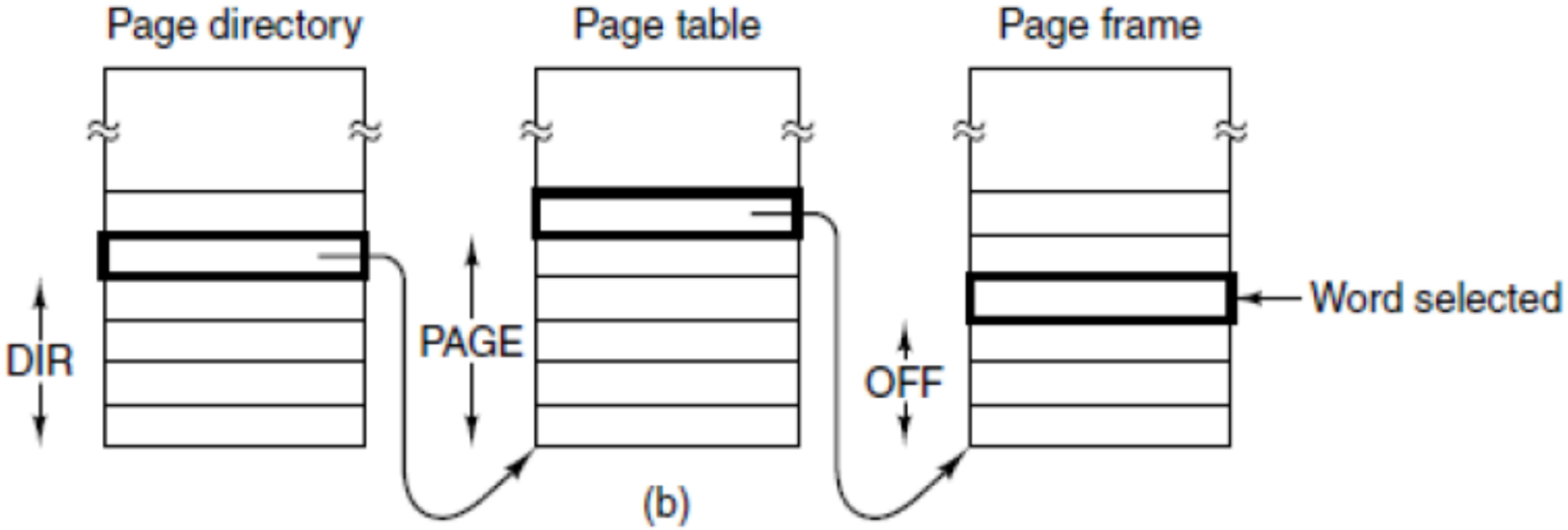


КАРХ: Тема_15: Архитектура Intel®64

Сегментна и странична преадресация.



(a)

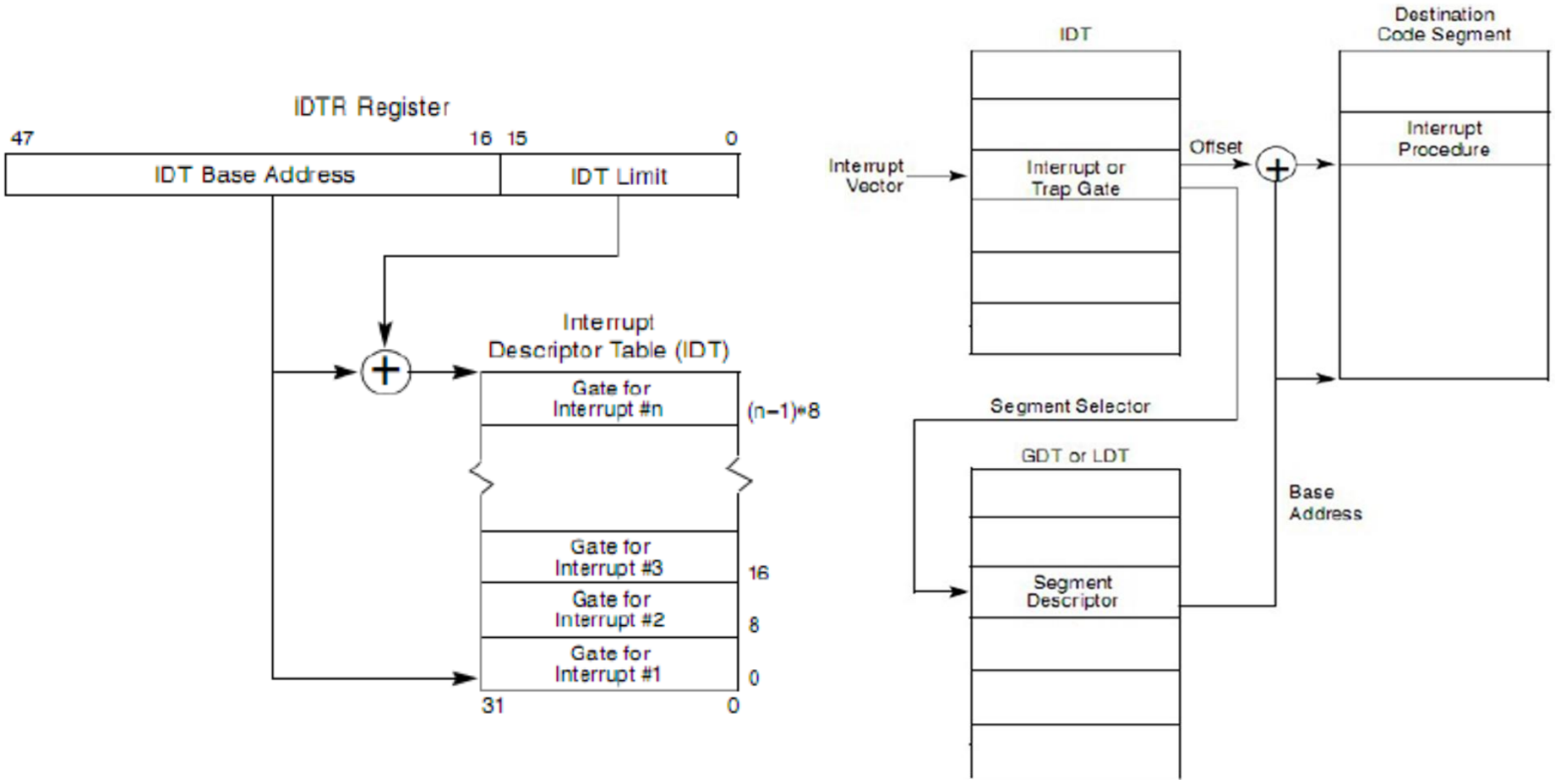


(b)

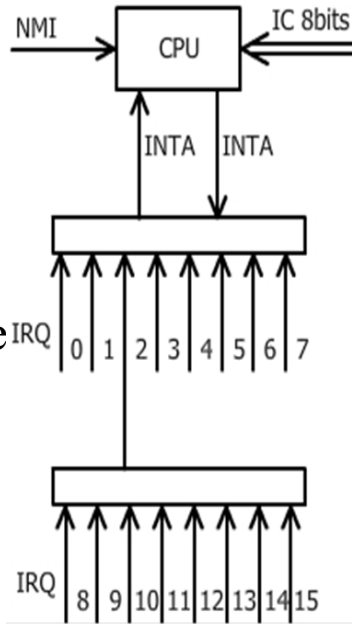
КАРХ: Тема_15: Архитектура Intel®64

Система за прекъсване – приоритети и обслужване.

- Намирането на съответната програма за обработване на дадено прекъсване става по следната схема:



КАРХ: Тема_15: Архитектура Intel®64



Система за прекъсване – приоритети и обслужване.

Контролер на прекъсванията

IRQ – Interrupt Request (0..15) – това са 16 линии за В/И прекъсвания. Първоначално линиите са били 8; след като се е наложило да се следи за повече прекъсвания, линия 2 се отделя за разширение и към нея по начина, показан на схемата, се включват нови 8 линии: IRQ 8 - 15. Редът на линиите определя техния приоритет – малък номер, голям приоритет. Като отчетем, че линия 2 е за разширение,приоритетите им са, както следва:

0 > 1 > [8 > 9 > 10 > 11 > 12 > 13 >14 > 15] > 3 > 4 > 5 > 6 > 7

IRQ0 – прекъсване при refresh на паметта.

IRQ1- прекъсване от клавиатурата.

IRQ8 – IRQ15 – прекъсвания от дискови устройства (това са по-важните прекъсвания).

IRQ3 – IRQ7 – прекъсвания на модеми, принтери и др.(по-бавни периферни устройства).

INTA – Interrupt Acknowledgement

След като се подаде заявка за прекъсване на ЦП, той изпраща отговор (INTA), ако е приел прекъсването и ще го обработи. Intel-ските процесори имат 1 вход за получаване на прекъсвания и 1 изход за потвърждаването им (показано на схемата). След като процесорът приеме прекъсването, се следва схема за обслужване на прекъсването по определен механизъм.

Код на прекъсване - Interrupt Code (IC)

След връщане на INTA сигнал устройството, на което е разрешено прекъсването, изпраща IC - 8-битов код на прекъсване. По този код ЦП търси програмата за обслужване на приетото прекъсване, от съответна заявка IRQ. В паметта се помнят 256 вектора на прекъсване. Всеки вектор съдържа адрес на програма за обработка.

NMI - Non-Maskable Interrupt.

КАРХ: Тема_15: Архитектура Intel®64

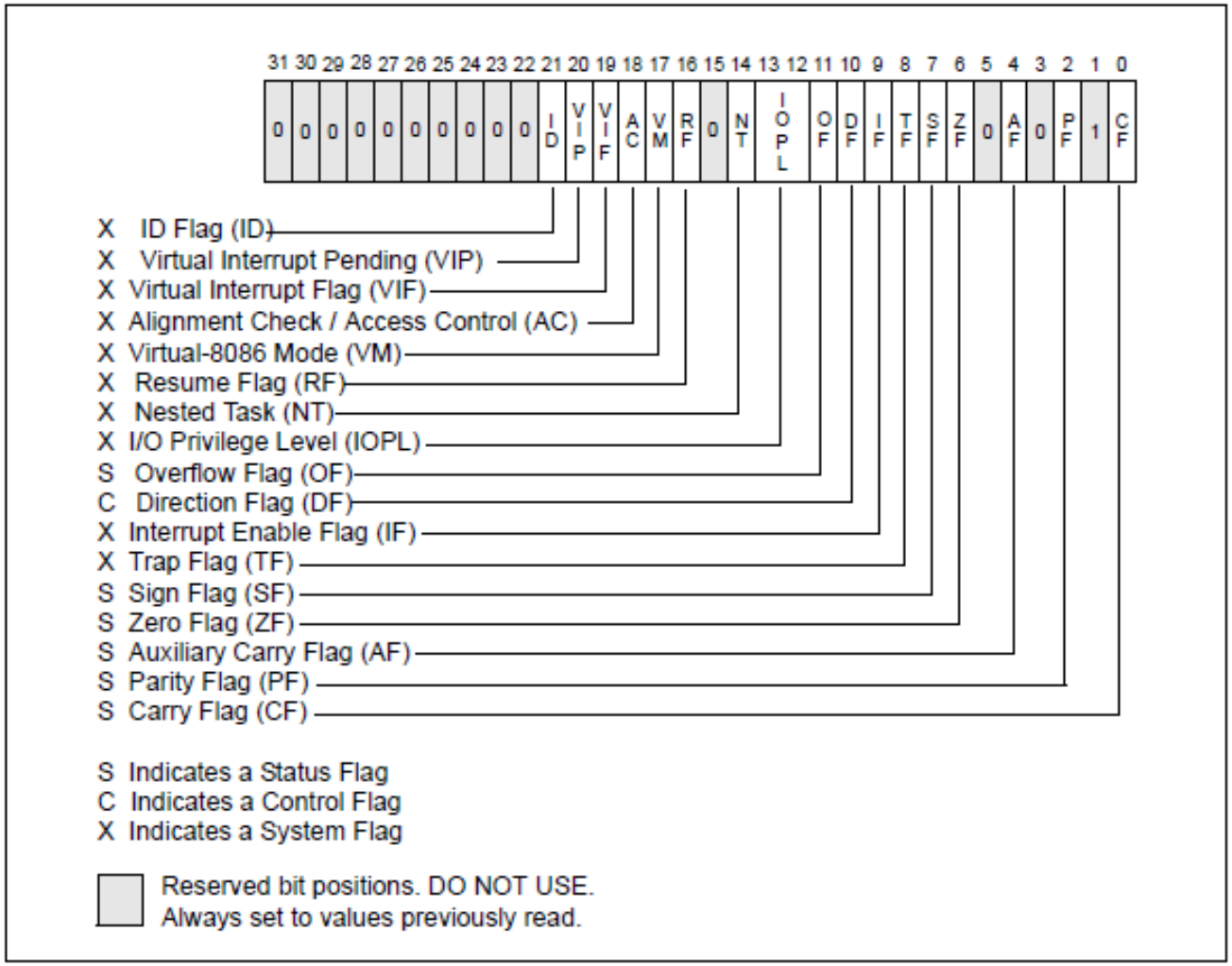
Сегментни регистри в 64 bit мод.

- В 64 bit мод регистрите CS, DS, SS, ES съдържат 0 базов адрес, което създава плоско адресно пространство за програмния код, данните и стека. FS и GS са изключение и могат да се използват като базисни при инструкции, използващи адресни операции.
- В 64 bit мод **не** се използва *сегментиране на адресното пространство*.

КАРХ: Тема_15: Архитектура Intel®64

EFLAGS регистър.

- 32 bit EFLAGS регистър съдържа 3 групи от флагове – **status flags**, **control flag** и **system flags**. Битове 1, 3, 5, 15, 22÷31 са запазени (не се използват).
- Някои от флаговете могат да бъдат променяни директно, но регистърът като цяло – не.
- При прекъсване на задача (в многозадачен режим) процесорът запазва автоматично състоянието на EFLAGS регистъра в *task state segment* (TSS) и зарежда от TSS EFLAGS регистъра със състоянието на флаговете на новата задача.



КАРХ: Тема_15: Архитектура Intel®64

EFLAGS регистър.

- При извикване на функция, прекъсване или при процедура за обработка на прекъсване, процесорът автоматично запазва състоянието на EFLAGS регистъра в процедурния стек.
- Статус флагове (**Status flags**)

CF (bit 0) – Carry flag – само този флаг *може да се променя програмно!*

PF (bit 2) – Parity flag – (става 1 при четен брой 1 в най-младшия байт на числото)

AF (bit 4) – Auxiliary Carry flag – (става 1 при пренос от/към 3-ти bit в BCD аритметика)

ZF (bit 6) – Zero flag

SF (bit 7) – Sign flag

OF (bit 11) – Overflow flag

- DF flag (bit 10) – Direction flag – използва се при работа със стрингове.
- В 64 bit мод EFLAGS регистъра се удължава до 64 bit и се нарича RFLAGS.

Старшите 32 bit са запазени, а младшите 32 bit са същите като в EFLAGS.

- В 64 bit мод EIP (instruction pointer) регистъра се удължава до 64 bit и се нарича RIP.

Благодаря Ви

за вниманието!

Литература

Основна литература.

1. David Money Harris, Sarah L. Harris, *Digital Design and Computer Architecture, Second Edition,* Morgan Kaufmann (MK) (Elsevier), 2013
2. Sarah L. Harris, David Money Harris, *Digital Design and Computer Architecture, ARM® Edition,* Morgan Kaufmann (MK) (Elsevier), 2016
3. 64-ia-32-architectures-software-developer-vol-1-manual (достъпно в мрежата)

Допълнителна литература.

1. David A. Patterson, John L. Hennessy, *Computer Organization and Design, The Hardware/Software Interface, ARM® Edition,* Morgan Kaufmann (MK) (Elsevier), 2017
2. David A. Patterson, John L. Hennessy, *Computer Organization and Design, The Hardware/Software Interface, RISC-V Edition,* Morgan Kaufmann (MK) (Elsevier), 2018
3. David A. Patterson, John L. Hennessy, *Computer Organization and Design, The Hardware/Software Interface, Fifth Edition,* Morgan Kaufmann (MK) (Elsevier), 2014 (достъпно в мрежата)
4. John L. Hennessy, David A. Patterson, *Computer Architecture, A Quantitative Approach, Fifth Edition,* Morgan Kaufmann (MK) (Elsevier), 2012 (достъпно в мрежата)
5. Andrew S. Tanenbaum, Todd Austin, *Structured Computer Organization, Sixth Edition,* PEARSON, 2013 (достъпно в мрежата)