Лекция 4: элементарная основа, производство и ошибки в процессорах.

Элементарная основа.

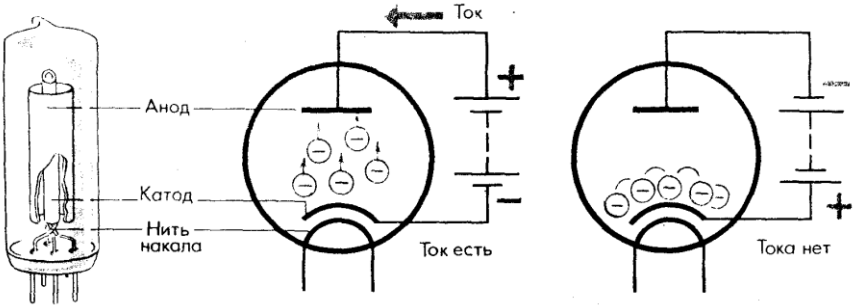
В первых компьютерах в качестве логического элемента использовались различные элементарные , от электрических, до механических и даже пневматических. Самым распространёнными были **электронные лампы**. Изобретателем считается Томас Эдисон. В 1883 году, в надежде улучшить своё основное изобретение — лампу накаливания он ввёл в вакуумное пространство лампы металлическую пластину с проводником, выведенным наружу. При экспериментах он заметил, что вакуум проводит ток, причём только в направлении от электрода к накалённой нити и только тогда, когда нить накалена. Это было неожиданно для того времени — считалось, что вакуум не может проводить ток, так как в нём нет носителей заряда. Изобретатель не понял тогда значение этого открытия, но на всякий случай запатентовал.
Далее конструкция дорабатывалась и в компьютерах использовались уже лампы, имеющие в составе три основных электрода

1. Анод – положительный электрод, используется для получения электронов с Катода и служит логическим элементом (именно его напряжение и принимается за 0 или 1)
2. Катод – источник электронов. Для эмиссии электронов катод нужно подогревать. Чаще всего катод делают из вольфрама.
3. Сетка – управляющий электрод, который служит для управления потоком электронов или устранением побочных явлений; располагается между катодом и анодом.

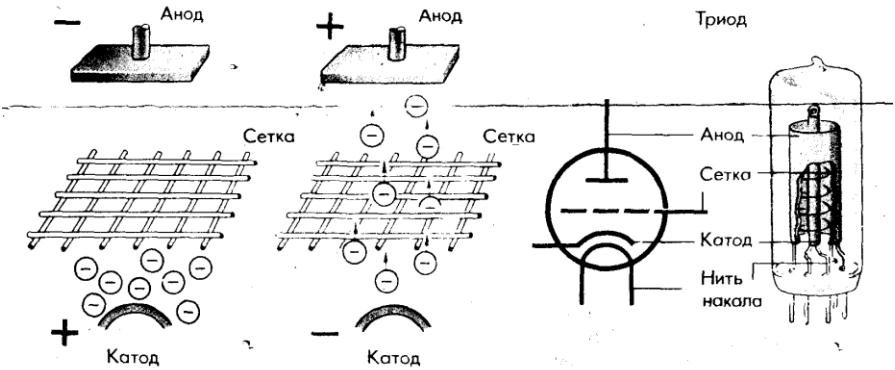
Фактически, изменяя напряжение на сетке можно менять интенсивности потока электронов, попадающих на анод.

4.

В зависимости от конструкции в лампе может быть до 7 сеток.



Лампа, эквивалентная диоду

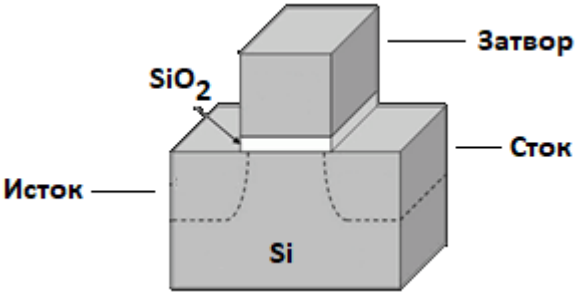


Лампа, эквивалентная триоду

В настоящее время лампы используются в военной и космической технике (ну и в акустике конечно же). Лампы имеют большую устойчивость к ионизирующему излучению, перепадам температур и электромагнитным помехам. Недостатками являются намного большие размеры и энергопотребление.

В настоящее время элементарной ячейкой процессора является **полевой транзистор**. По сравнению с электронной лампой он имеет

ряд преимуществ, при аналогичной функциональности.

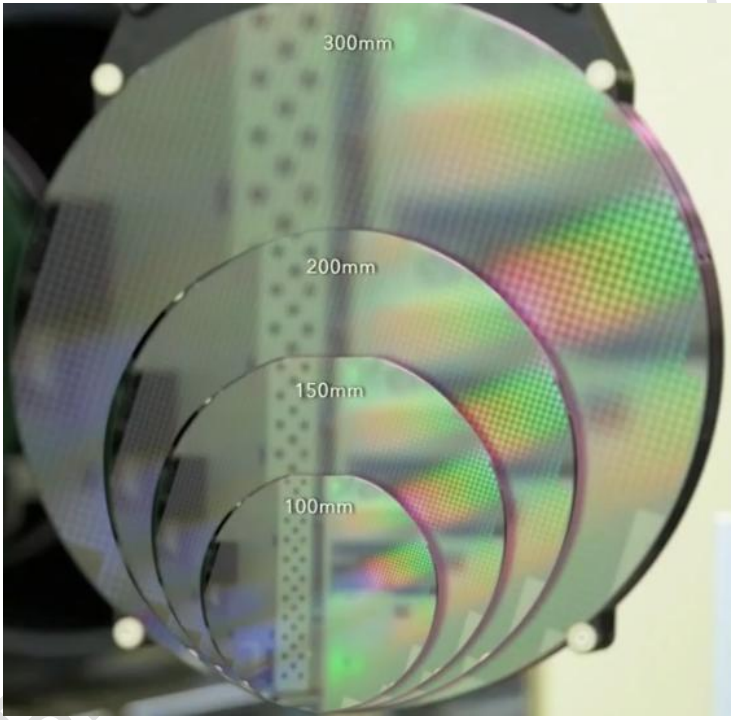


Преимуществами являются:

- Размеры и потребление тока
- Отсутствие требующего подогрева катода
- Сток и исток взаимозаменяемы, то есть их можно функционально поменять местами
- Существуют транзисторы как с каналом из полупроводника n типа, так и p типа, что позволяет создавать комплементарные (связанные) пары транзисторов

Первые полупроводниковые изделия были из германия и первые транзисторы изготавливались из него же. Но как только начали делать полевые транзисторы (под затвором которого находится специальный изолирующий слой — тонкая диэлектрическая пленка, управляющая «включением» и «выключением» транзистора), германий тут же заменили на кремний. Давайте по подробнее разберемся почему именно кремний.
Во первых, кремний второй по распространенности элемент в земной коре, то есть этого сырья просто много. Обычный песок состоит в основном из диоксида кремний SiO_2. Разумеется, что для производства микросхем требуется сверхчистое сырье и песок с соседнего пляжа скорее всего не подходит. Найти чистое сырье все

таки не так и сложно. Для восстановления кремния из оксида его смешивают с углеродом и нагревают до температуры в 1000°С . Получившееся сырье и есть чистый, монокристаллический кремний, с чистотой около 100% (не более 1 атома на 10.000.000 атомов кремния). Сырье формируется виде цилиндрического монокристалла, диаметром от 100мм до 300мм.

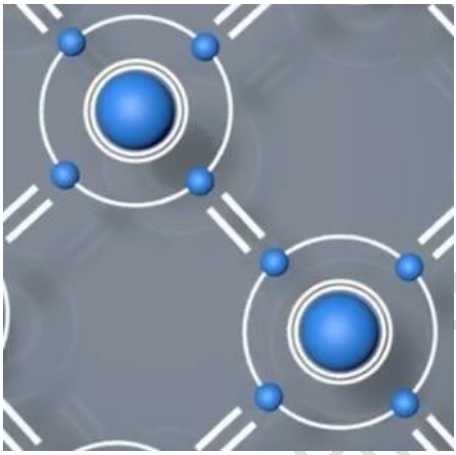


Чем больше диаметр, тем больше на будущей пластине разместится готовых процессоров и, соответственно, себестоимость одного процессора будет ниже.

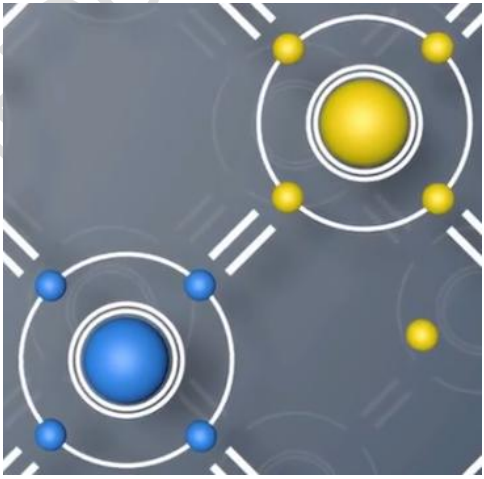
Во вторых, чистый кремний представляет собой полупроводник, то есть при определенных условиях он ведет себя как проводник, а при других условиях - как изолятор.

Это свойство связанно с атомной структурой кремния : у каждого

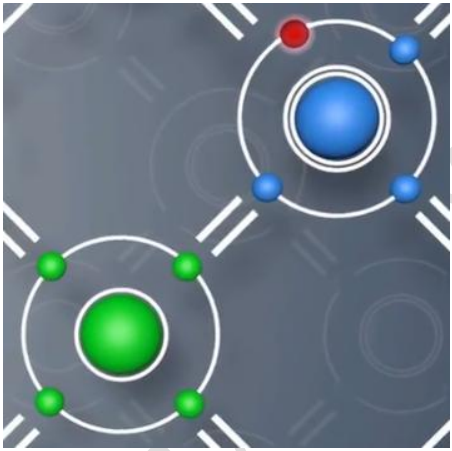
атома есть 4 электрона, которые полностью использованы при построении атомной решетки. Свободных электронов для передачи заряда нет.



То есть чистый кремний является изолятором. Для изменения его свойств в кремний добавляют атомы другого вещества. Наиболее подходящими являются Бор и Фосфор.



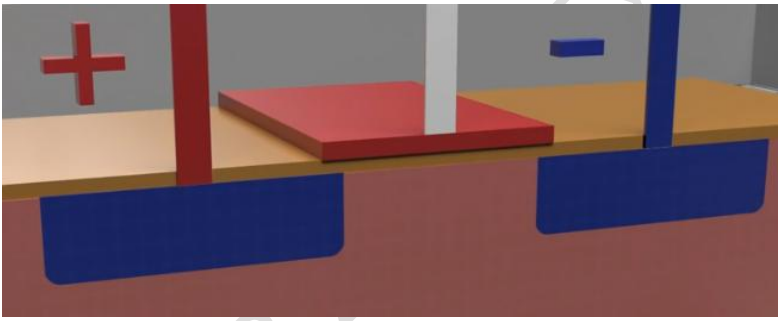
Фосфор имеет 5 электронов и при помещении в решетку кремния получается, что один из электронов «лишний» и может свободно перемещаться в кристаллической решетке.
Получается **полупроводник n-типа**.
Бор наоборот имеет только три электрона.



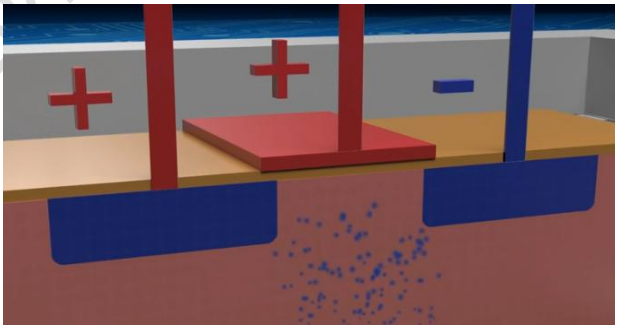
Получаются электронные «дырки». Вместе с примесью бора кремний образует полупроводник **p-типа**.
Соединение SiO_2 является **изолятором** .
Наличие двух разных типов полупроводников и изолятора позволяет создавать полевые транзисторы – основные элементы современных чипов. Основное назначение транзистора служить управляющим вентилем для электрического тока – пропускать или не пропускать заряд в зависимости от внешних условий. Сам принцип действия транзистора основан на свойстве полупроводникового материала менять свою способность проводить ток под действием внешнего фактора.
Приведем одну из возможных схем полевого транзистора. В основе транзистора лежит слой полупроводника p-типа. В нём размещены две области полупроводника n-типа. Сверху поверхность покрыта изолятором. Над промежутком между

областями n-типа расположен проводник. К обеим областям подходят свои проводники.
Проводники полевого транзистора имеют следующие названия:

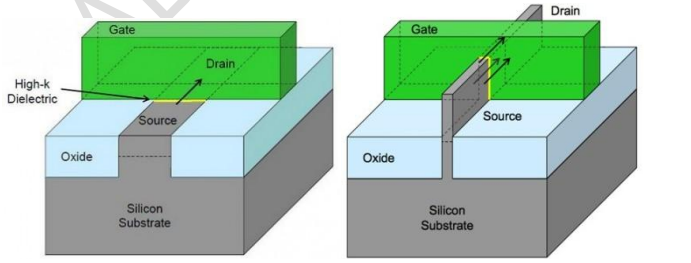
- исток (англ. source) — электрод, из которого в канал входят основные носители заряда;
- сток (англ. drain) — электрод, через который из канала уходят основные носители заряда;
- затвор (англ. gate) — электрод, служащий для регулирования поперечного сечения канала.



Принцип действия прост — при отсутствии напряжения на затворе, слой полупроводника между стоком и истоком проявляет свойства изолятора и не пропускает ток.



При появлении напряжения на затворе электроны из слоя подложки притягиваются электромагнитным полем и скапливаются около слоя изолятора напротив затвора. Это создает область проводимости и позволяет протекать заряду из истока в сток.
Это очень упрощенное представление о схеме работы транзистора и имеется очень много физических нюансов его работы, но это тема не курса архитектура компьютера, а курса радиофизики. Для нас важен лишь принцип – в зависимости от тока на затворе транзистор либо проводит ток, либо нет. Наличие или отсутствие тока может быть интерпретировано как логические 0 и 1. На этом свойстве и основано использование транзистора, как основы логических схем.
Полевой транзистор удобен тем, что его легко получить методом трафаретной печати, однако классическая схема транзистора имеет физические ограничения по минимальным размерам. 24 сентября 2008 года сотрудники университета Рочестера и Массачусетского технологического институт представили технологию, называющуюся **true3D**: первый трехмерный процессор, работающий на частоте 1,4 ГГц. До этого момента все процессоры по сути были «плоскими». Переход к трехмерным схемам позволяет существенно уплотнить компоновку процессора и значит увеличить его производительность.
Intel представила технологию tri-gate в 2011 году в процессорах Ivy Bridge, хотя её разработка началась в 2002 году.



Слева «классический» полевой транзистор, справа – tri-gate транзистор. Как видим, несмотря на громкие заявления, схема остается плоской, просто освоена «печать» транзисторов с вертикально расположенным стоком и истоком. Это позволяет уплотнить компоновку схемы, но до полноценного 3D ещё далеко. Аналоги от Samsung и HiSilicon представлены в том же 2011году, а AMD и NVidia вышли в 2016 году.

Вообще, при переходе на более «тонкий» тех процесс все составляющие части транзистора уменьшаются пропорционально. В чипах с техпроцессом в 65нм толщина слоя диэлектрика затвора из SiO_2 составляла порядка 1.2 нм, что эквивалентно пяти атомарным слоям. Фактически, это физический предел для данного материала, поскольку в результате дальнейшего уменьшения самого транзистора (а значит и уменьшения слоя диоксида кремния), ток утечки через диэлектрик затвора значительно возрастает, что приводит к существенным потерям тока и избыточному тепловыделению. При переходе на 45нм компания Intel стала использовать новый материал, так называемый high-k диэлектрик, который заменил бесперспективно тонкий слой диоксида кремния. Слой на базе окиси редкоземельного металла гафния стал более толстым, но это позволило сократить ток утечки более чем в десять раз, сохранив при этом возможность корректно и стабильно управлять работой транзистора. Новый диэлектрик оказался плохо совместим с затвором из поликремния, но и это не стало препятствием — для повышения быстродействия затвор в новых транзисторах был выполнен из металла. Например, процесс 22 нм производится по такой же технологии. Для перехода на более тонкие тех процессы скорее всего потребуется уже заменять подложку из кремния на что-то более совершенное: арсенид галия или вообще на графен.

Не надо думать, что с переходом нескольких фабрик на 32нм, все вдруг станет производиться по этому техпроцессу – тем же чипсетам и другим периферийным схемам это просто не нужно – в

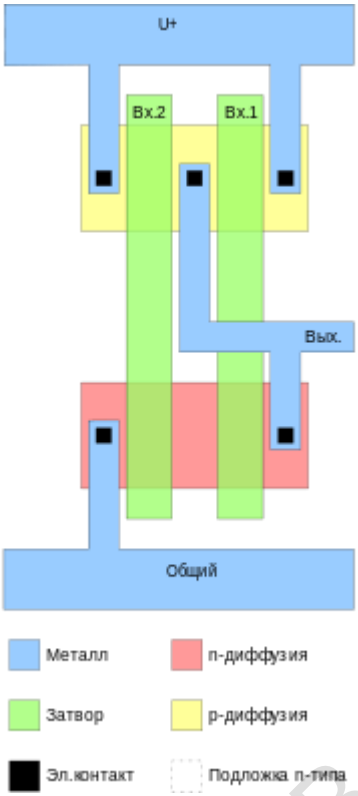
большинстве случаев в них используется 45нм. Так же многое решает и архитектура процессора — отечественнные процессоры Байкал до недавнего времени производились в Зеленограде по 65нм процессу (сейчас заказы на производство размещаются на Тайване по более совершенным тех процессам). Производительности же их вполне достаточно для сетевого оборудования и промышленных контроллеров.

Пример реализации логического элемента.

На основе транзистора можно реализовать все логические операции. Рассмотрим реализацию операции И-НЕ.

A	B	Результат
0	**0**	**1**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**

В данной схеме на выходе 0 только когда на обоих входах 1. В виде полевых транзисторов эта схема реализуется так:

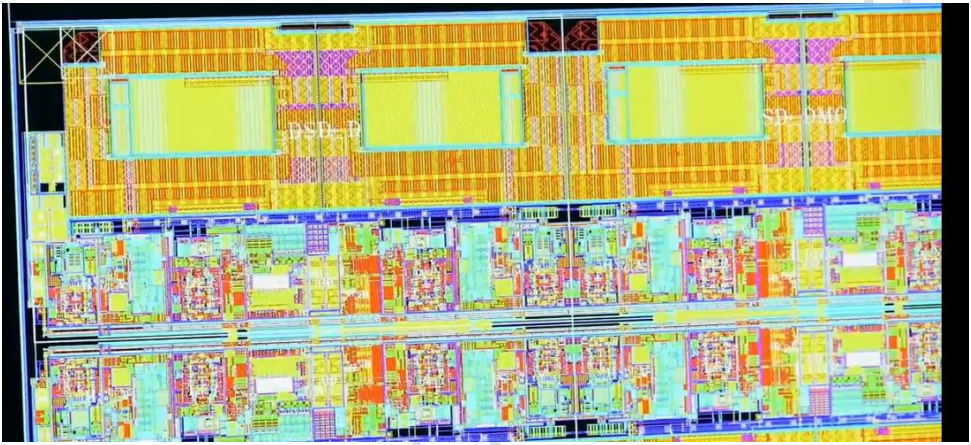


Не вдаваясь в физические тонкости процесса получаем, что схема И-НЕ собирается из двух взаимосвязанных полевых транзисторов, в основе которых лежат полупроводниковые пары разных типов. При наличии напряжения на обоих затворах (то есть при сигнале A=1 B=1) каналы обоих транзисторах изолируются и на выходной проводник заряд не попадает.

Производство.

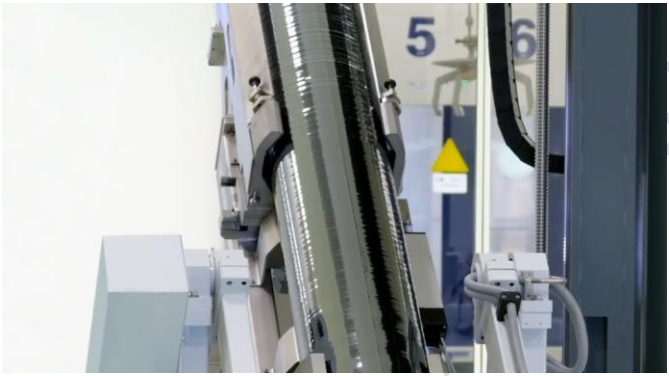
Технически ,процессор представляет собой одну сверхбольшую интегральную схему, состоящую из нескольких миллиардов элементов. Основным элементов является полевой транзистор. В

настоящее время размеры полевых транзисторов измеряются нанометрами (10^{-9} м).
Производство начинается с создания схемы в специальных программах. Над разработкой схемы трудятся сотни высококвалифицированных специалистов.

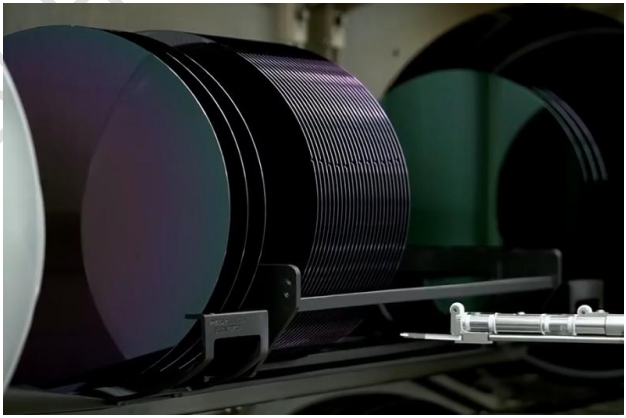


После окончания разработки схемы начинается собственно производственный процесс, который проходит в специальных чистых помещениях.

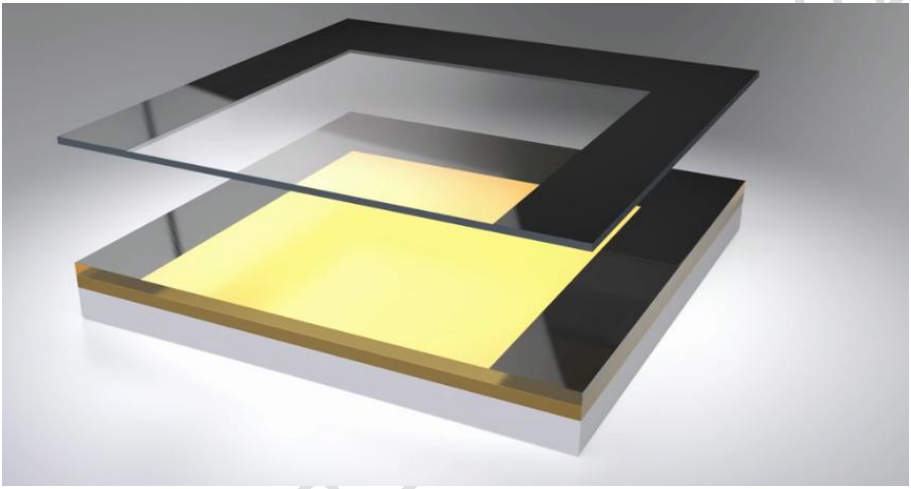
Производство процессоров происходит на **пластинах** из кремния. Цикл производства одной пластины может достигать 3 месяцев. Все начинается с небольшого кристалла (который и опускают в расплав очищаемого кремния) – позже он превращается в специальный монокристаллический «буль» ростом с человека.



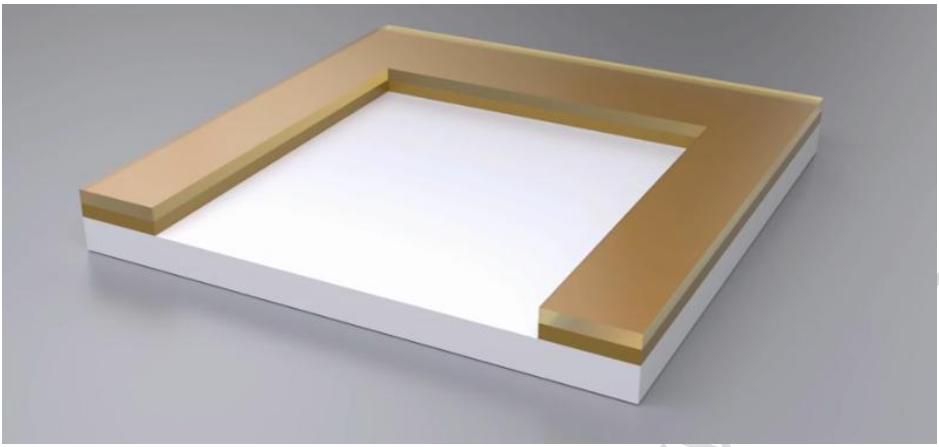
Далее убираются основные дефекты и специальными нитями (с алмазным порошком) буль нарезается на диски – каждый диск тщательно обрабатывается до абсолютно ровной и гладкой (на атомарном уровне) поверхности. Толщина каждой пластины около 1мм.



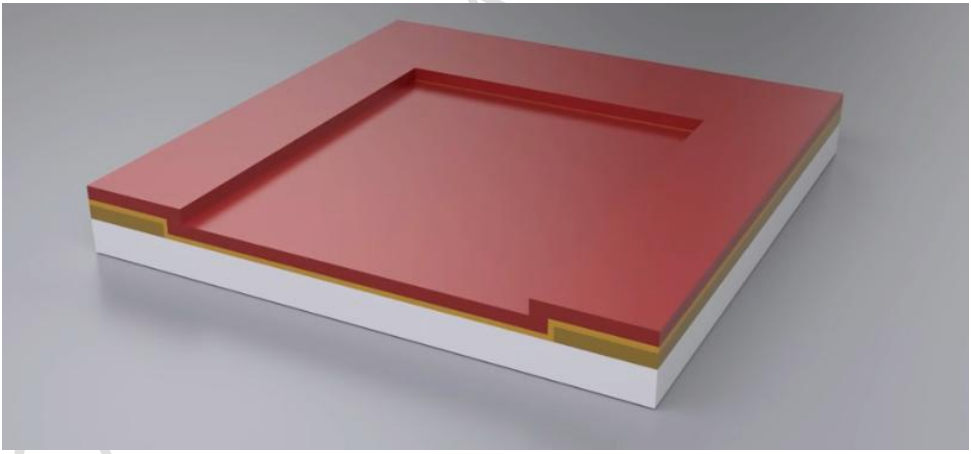
Далее поверхность пластины покрывается оксидом кремния, то есть изолятора. После на пластину наносят (а точнее - наливают и разравнивают по действием центробежной силы) специальный состав – фоторезист.
Далее в специальном аппарате – степпере –слой через маску облучают излучением строго определенной длины волны.



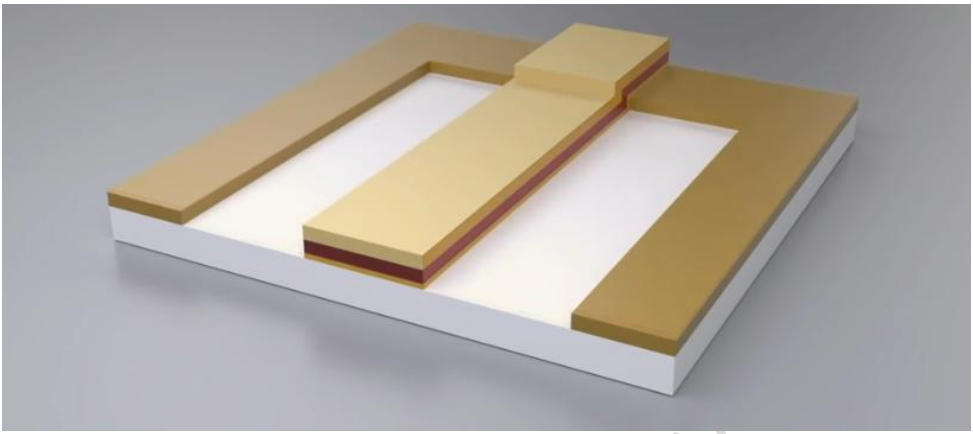
Маски изготавливают при проектировании процессора и используют для формирования рисунков схем в каждом слое процессора. Под воздействием излучения засвеченные участки фотослоя становятся растворимыми, и их удаляют с помощью растворителя (плавиковая кислота), открывая находящийся под ними диоксид кремния. Открытый диоксид кремния удаляют с помощью процесса, который называется "травлением". Затем убирают оставшийся фотослой, в результате чего на полупроводниковой пластине остается рисунок из диоксида кремния. На этом этапе используется либо химический метод (старый способ), либо ультразвуковая и/или лазерная чистка в среде инертного газа.



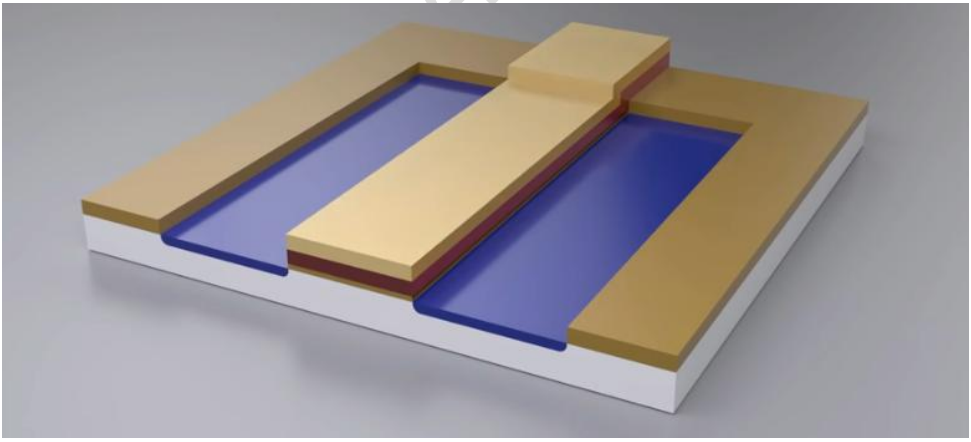
После этого выполняется снова нанесение изолятора (оранжевый слой на слайде), а далее слоя поликремния (красный слой) для создания затвора.



Далее повторяются этапы нанесения фоторезиста, этап облучения через маску (другую уже разумеется), смытие незасвеченного фоторезиста, слоя полисиликона и слоя изолятора.

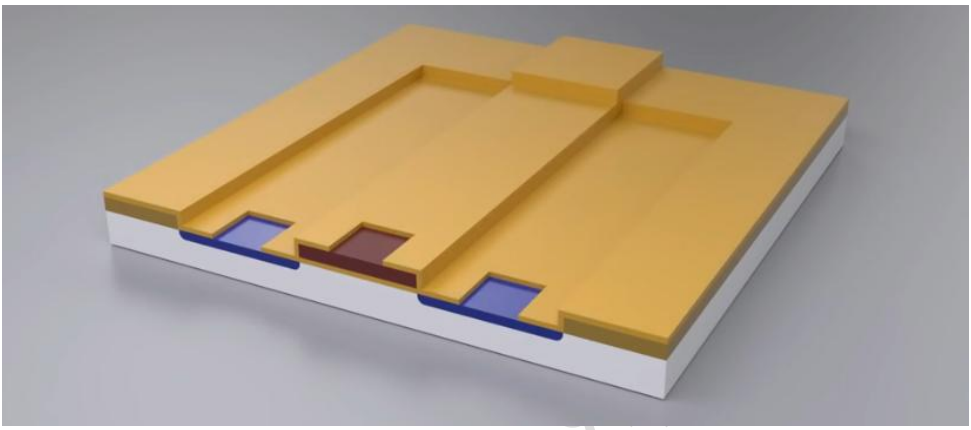


Таким образом формируется затвор.
В ходе следующей операции, называемой "легированием", открытые участки кремниевой пластины бомбардируют ионами требуемых химических элементов, которые формируют в кремнии области p-типа и n-типа, в зависимости от требований разработанной схемы.

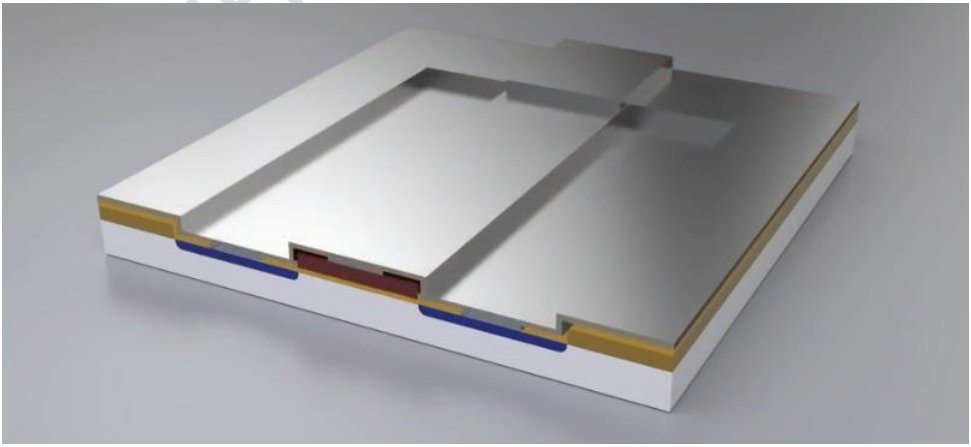


Далее опять повторяется снятие фоторезиста, напыление оксида, следующего слоя фоторезиста и облучение через следующую маску.

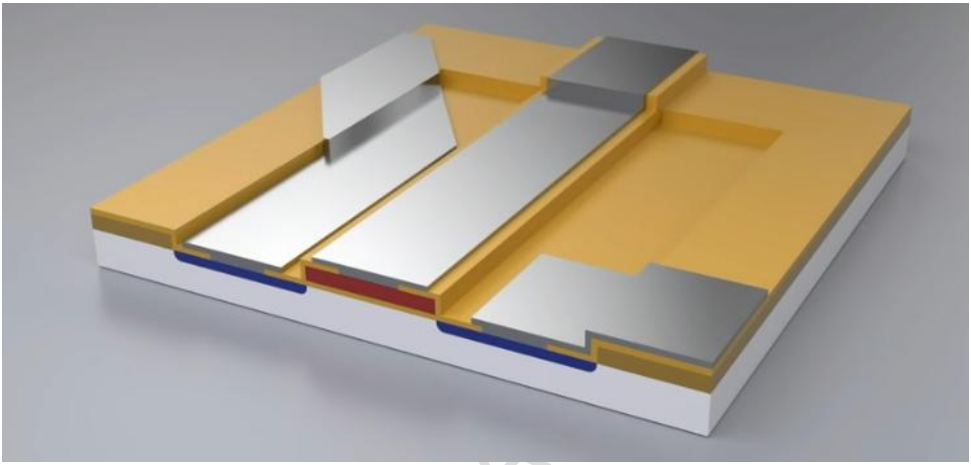
Далее смывается засвеченный фоторезист, изолятор и незасвеченный фоторезист. Получаются «окна» для подсоединения проводников к элементам транзистора.



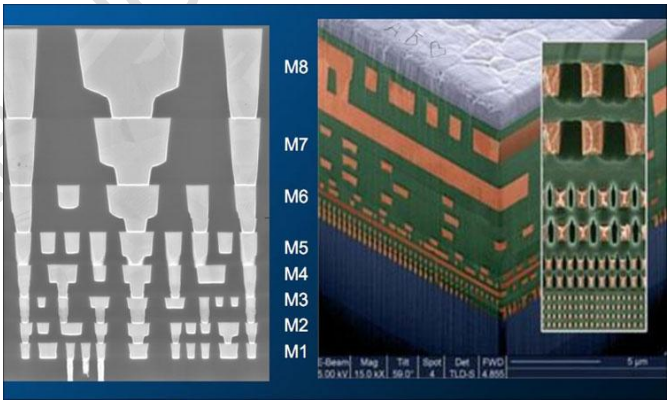
Поверхность легируется металлом для создания проводников. В 0.13-микронном технологическом процессе корпорация Intel применила медные проводники. В 0.18-микронном производственном процессе и процессах предыдущих поколений Intel применяла алюминий. И медь, и алюминий - отличные проводники электричества.



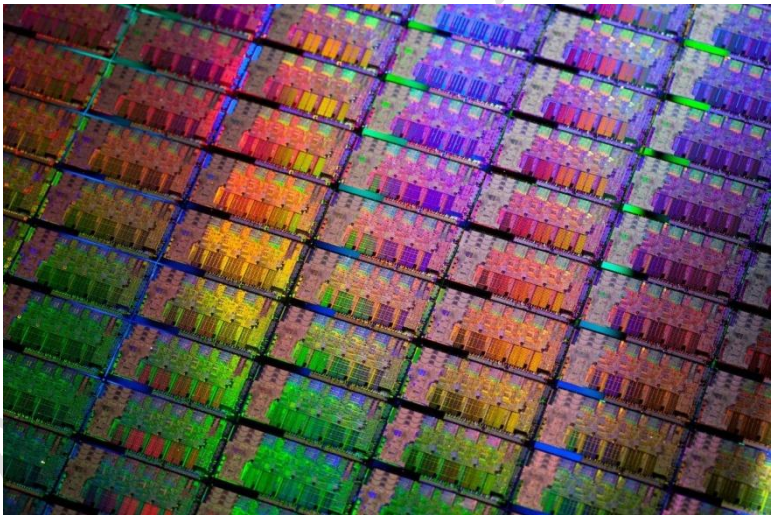
Далее опять наносится фоторезист, производится экспонирование через маску, смывается незасвеченый фоторезист, освободившийся металл (останется только металл под фоторезистом) и наконец смывается засвеченный фоторезист.



Таким образом получаем «напечатанный» полевой транзистор. Операции повторяются множество раз для формирования всей разработанной схемы.



Каждый слой процессора имеет свой собственный рисунок, в совокупности все эти слои образуют трехмерную электронную схему. Нанесение слоев повторяют 20 - 25 раз в течение нескольких недель.
Чтобы выдержать воздействия, которым подвергаются подложки в процессе нанесения слоев, кремниевые пластины изначально должны быть достаточно толстыми. Поэтому прежде чем разрезать пластину на отдельные микропроцессоры, ее толщину с помощью специальных процессов уменьшают на 33% и удаляют загрязнения с обратной стороны. Затем на обратную сторону "похудевшей" пластины наносят слой специального материала, который улучшает последующее крепление кристалла к корпусу. Кроме того, этот слой обеспечивает электрический контакт между задней поверхностью интегральной схемы и корпусом после сборки.



Готовая пластина с кристалами

После этого пластины **тестируют**, чтобы проверить качество выполнения всех операций обработки. Чтобы определить, правильно ли работают процессоры, проверяют их отдельные

компоненты. Если обнаруживаются неисправности, данные о них анализируют, чтобы понять, на каком этапе обработки возник сбой. Затем к каждому процессору подключают электрические зонды и подают питание. Процессоры тестируются компьютером, который определяет, удовлетворяют ли характеристики изготовленных процессоров заданным требованиям.
После тестирования пластины отправляются в сборочное производство, где их разрезают на маленькие прямоугольники, каждый из которых содержит интегральную схему. Для разделения пластины используют специальную прецизионную пилу. Неработающие кристаллы отбраковываются. Далее кристалл **упаковывается** в корпус процессора.

Сверху находится защитная металлическая крышка, которая помимо защитной функции, так же выполняет роль теплораспределителя. Под теплораспределителем находится сам чип процессора. Еще ниже – специальная подложка, которая нужна для разводки контактов (и увеличения площади «ножек»), чтобы процессор можно было установить в сокет материнской платы.

Ошибки в процессорах.

Как и в любом устройстве, в процессорах могут быть ошибки. Ошибки были двух типов

- Аппаратные
- Архитектурные

Аппаратные ошибки представляют собой недоработки в схемотехнике кристалла процессора, а позже – в микрокоде самого процессора. Они являются следствием ошибки инженера и устраняются в последующих продуктах, или прошивках к ним. Другое дело ошибки архитектурные. Они содержатся в логике создания процессора и часто не подлежат исправлению в рамках текущей архитектуры целой линейки процессоров, требуя замены основ построения кристаллов. Скорее всего, что исправление таких ошибок приведет к полной неработоспособности уже имеющегося ПО, в том числе и прикладного.

Аппаратные ошибки в CPU.

Первая из широко известных ошибок в процессорах был обнаруженная в 1994 году ошибка **Pentium FDIV** в математическом сопроцессоре в Intel Pentium .

Ошибка выражалась в том, что при проведении деления над числами с плавающей запятой при помощи команды процессора FDIV в некоторых случаях результат мог быть некорректным. Согласно заявлению Intel, причиной проблемы послужили неточности в таблице поиска, используемой при проведении операции деления.

Как оказалось, в Intel знали об этой проблеме, но молчали. К тому же в Intel считали, что, поскольку этот дефект существенен лишь для узкого круга пользователей (математиков и других ученых), то пользователи, которые хотят заменить процессор, должны обратиться в компанию и доказать, что именно им эта замена

необходима. Стремление производителя утаить проблему и реакция на её обнаружение вызвали недовольство потребителей и обширную критику в СМИ, в том числе жесткий репортаж CNN. В результате компания изменила позицию и объявила, что будет свободно обменивать дефектные процессоры всем желающим. Энди Гроув принес публичные извинения. История стоила Intel более половины прибыли за последний квартал 1994 г. — $475 млн..
Воспроизвести ошибку можно самостоятельно, произведя умножение и деление на одно и тоже число. Например:

4195835*3145727/3145727 = 4195579

хотя получаться, конечно, должно исходное число.
Исправление самого процессора не возможно, ошибка содержится не в микрокоде процессора, а в аппаратной реализации. Возможно лишь программно обойти ошибку. Например, в компиляторах была предусмотрена проверка на наличие этой ошибки и вызов заменяющей процедуры в случае обнаружения ошибки.

Следующей по времени обнаружения была **ошибка FOOF**.
F0 0F C7 C8 — последовательность байтов, формирующих недействительную машинную команду процессоров семейства x86. В процессорах Pentium MMX и Pentium OverDrive, вследствие аппаратной недоработки, команда, будучи выполненной на любом уровне привилегий, приводила к мёртвому зависанию процессора, что отрицательно сказывалось на надёжности системы в целом. Инструкция представляет собой команду:

lock cmpxchg8b eax

Операндом может быть любой другой регистр помимо eax.
cmpxchg8b используется для сравнения содержимого пары

регистров eax и edx с 8 байтами содержимого некоторого участка памяти. При этом происходит попытка поместить 8-байтовый результат в 4-байтный регистр.
Сама по себе эта команда просто вызывает исключение, однако при сочетании с префиксом lock (он используется для предотвращения обращения двух процессоров к одному и тому же участку памяти одновременно) обработчик исключений не вызывается, процессор перестает обрабатывать прерывания и для приведения его в рабочее состояние требуется перезагрузка.
Ошибка была исправлена на аппаратном уровне в процессоре PentiumPro и более поздних. Замену уязвимых процессоров Intel произвела бесплатно. Для уязвимых процессоров в различных ОС были выпущены патчи, запрещающие указанную инструкцию.

В 2014 году в процессорах поколения Haswell и ранних образцах Broadwell (четвертое и пятое поколение) существует **ошибка поддержки транзакционной памяти**. TSX инструкции, позволявшие существенно ускорить работу с блоками памяти, содержали ошибки, вызывающие потерю содержимого памяти. В более поздних процессорах инструкции работают корректно, для уязвимых процессоров обновлением прошивки микропроцессора инструкции были отключены. Отзывной компании не проводили.
В начале 2016 года была обнаружена ошибка в процессорах Intel Core шестого поколения: при сложных математических вычислениях процессор «зависает». Баг проявился в ходе работ по поиску простых чисел Мерсенна с помощью инструмента Prime95, хотя на остальных поколенияхчипов Intel проект работает стабильно. К примеру, сбой системы был обнаружен при работе с **экспонентой степени 14 942 209**. Intel сообщила о разработке патча для системы BIOS, устраняющего указанную проблему. Отзывной компании опять не проводилось.

Не только Intel сталкивается с ошибками в своих процессорах. AMD так же допускает «промахи» в своих продуктах. Например, в 2017 году в процессорах Ryzen, основанных на кристаллах Zeppelin стэпинга B1 содержалась **ошибка сегментации**, с которой эпизодически сталкивались пользователи различных Unix-подобных систем на основе Ryzen (например, в Gentoo Linux или FreeBSD). Суть проблемы заключалась в том, что при продолжительных многопоточных тяжёлых нагрузках, например, при компиляции масштабных проектов, могла возникнуть ошибка сегментации, спровоцированная неправильной работой процессора. В начале августа эта проблема была признана AMD и охарактеризована как «сложная маргинальная проблема, возникающая исключительно при определённых нагрузках в Linux». При этом было подтверждено, что она затрагивает только процессоры Ryzen, в то время как вышедшие в более поздние сроки процессоры EPYC и Ryzen Threadripper ей не подвержены. Тем не менее, AMD пообещала уделить внимание полному исправлению данной ошибки, и по всей видимости это и было сделано.Поставляемые с августа 2017 года процессоры семейства Ryzen уже не содержат проблемы, выливающейся в возникновение ошибок сегментации. Починка процессоров на самом деле произошла где-то в середине июня, и те чипы, которые произведены после 25 недели 2017 года, не должны вызывать никаких ошибок. Время производства процессора закодировано в

маркировке CPU в строке, начинающейся с символов UA. Следующие за ними четыре цифры – это год и номер недели производства. Соответственно процессоры, не вызывающие ошибок сегментации в Linux, должны иметь в этом месте маркировки число, превышающее 1725. Никаких же иных отличительных признаков у исправленных Ryzen нет: при программной идентификации они так же, как и их предшественники, демонстрируют номер степпинга B1. Скорее всего, что исправление выполнено виде обновления микрокода процессора, а не на аппаратном уровне. Но никаких комментариев от AMD более не поступало.

Архитектурные ошибки.

Более существенными уязвимостями являются ошибки, связанные с архитектурными особенностями современных процессоров. Этим ошибкам подвержены все процессоры, использующие эти особенности и качество исполнения самих процессоров не влияет на наличие или отсутствие этой ошибки. Наиболее известны две

- Spectre
- Meltdown

Spectre – название уязвимости в современных CPU со спекулятивным выполнением команд.

Уязвимость была обнаружена исследователями североамериканской корпорации и группой сотрудников Грацского

технического университета. Уязвимость была найдена в середине 2017 года и несколько месяцев находилась на стадии закрытого обсуждения и исправления. Публикация подробной информации и исправлений была запланирована на 9 января 2018 года, но детали уязвимости были обнародованы 4 января 2018 года одновременно с атакой Meltdown, из-за публикаций журналистов, которые узнали об исправлениях для борьбы с Meltdown из списка рассылки ядра Linux.

В основе уязвимости лежит спекулятивное выполнение команд и использование общей кэш памяти несколькими процессорными ядрами. Затрагивает большинство современных микропроцессоров, в частности, архитектур x86/x86_64 (Intel и AMD) и некоторые процессорные ядра ARM]. Уязвимость потенциально позволяет локальным получить доступ к содержимому виртуальной памяти текущего приложения или других программ, то есть нарушить защиту, изоляцию приложений друг от друга, в том числе и ядра ОС. Угрозе присвоены два CVE-идентификатора:

- CVE-2017-5753
- CVE-2017-5715

Так же имеется вариант реализации атаки через Java-script для получения данных из других вкладок браузера. Особенно актуальна в виду распространенности систем ДБО (Дистанционного Банковского Обслуживания) с использованием Java-апплетов. Реализация использует **неверное предсказание условных и косвенных переходов**. Например

If(x< array1.size)
Y=array2[array1[x]*256];

При неверном предсказании перехода будет спекулятивно выполнено присваивание переменной, например получится реализация переполнения стэка или области памяти. Можно

подобрать такие значения массивов, что выбранный «x» позволит обратится к области памяти другого процесса и получить значения, которые защищены от обращения из приложения на аппаратном уровне.
Косвенный переход может использовать для ветвления больше, чем два адреса. Например, инструкции процессоров семейства x86 могут выполнять переход, используя значение адреса в регистре (jmp eax), в памяти (jmp [eax], или jmp dword ptr [0xdeadc0de]), или в стеке (ret). Инструкции косвенных переходов имеются также в процессорах ARM (mov pc, r14), MIPS (jr $ra), SPARC (jmpl %o7), RISC-V (jarl x0,x1,0), и многих других.
Если определение адреса косвенного перехода откладывается из-за промаха кэша, и предсказатель косвенных переходов «натренирован» с использованием специально подобранных адресов, может произойти спекулятивное исполнение команд по адресу, заданному злоумышленником. Команд, которые иначе никогда бы не были выполнены.
На август 2018 года **не существовало** готовых программных решений для устранения уязвимости. В ранних версиях уведомления CVE содержалось предложение о полной замене всех уязвимых процессоров. В более поздних уведомлениях это предложение отсутствует, наличествуют лишь

1. Предложение обновления микрокода процессора и добавление новых, «безопасных» команд
2. Исправление компиляторов с устранением потенциально опасных инструкций

Meltdown - технически аналогичная уязвимость, однако направлена она на получение данных из ядра ОС.

Возможность атаки порождается тремя механизмами, позволяющими ускорить работу процессора, причём каждый из этих механизмов по отдельности не создаёт уязвимости:

1. Глубокое **спекулятивное выполнение операций**, в том числе чтения из оперативной памяти без проверки прав доступа процесса к читаемым областям. Если в итоге спекулятивное исполнение будет признано ошибочным, то исключение по доступу к запретной области памяти не генерируется, а результаты загрузки данных в регистры просто отменяются.
2. **Отсутствия очистки кэша** от результатов ошибочного спекулятивного исполнения (подобная очистка, вероятно, снизила бы скорость работы процессора). Формально содержимое кэша недоступно программе напрямую; но анализ времени доступа к отдельным ячейкам оперативной памяти может косвенно указать, есть ли конкретные данные в кэше или нет (в данном случае — были ли эти данные загружены при спекулятивном выполнении команд).
3. **Ядро** операционной системы **держит свои данные в адресном пространстве процесса**, защищая их от доступа уровнем привилегий. Данная технология позволяет быстрее исполнять системные вызовы. При таких вызовах повышается уровень привилегий, а при возврате обратно уровень привилегий снова понижается, при этом не требуется перезагружать таблицу страничных дескрипторов.

Именно третья особенность и позволяет аналогично атаке Spectre получать данные из ядра ОС. Атака может быть проведена

примерно следующим образом.Для того, чтобы прочитать бит 0 из защищённой области памяти Ap, атакующий:

1. Очищает кеш для адресов A0u и A1u (из адресного пространства атакующего, доступного для чтения/записи)
2. Выполняет ветвление по условию, заведомо известному атакующему
3. В ветке кода, которая, по условию, никогда выполняться не должна (но будет выполнена при спекулятивном выполнении):
4. Читает значение V(Ap) из защищённой области памяти по адресу Ap
5. Посредством выполнения побитовой операции над значением V(Ap) получает адрес A0u либо A1u
6. Читает память по полученному адресу (A0u или A1u)
7. При обычном выполнении шаг 4 вызывает ошибку защиты, однако на этапе спекулятивного выполнения на уязвимых архитектурах эта ошибка временно игнорируется, продолжая выполнение шагов 5 и 6. В результате в кеш загружается одно из значений — с адреса A0u или A1u. Выяснив условие ветвления, процессор аннулирует все результаты выполнения шагов 4, 5 и 6, но состояние кеша остается неизменным.
8. После этого атакующему достаточно прочитать «свои» адреса A0u и A1u, измеряя время доступа к ним. И на основании замеров определить, какой бит (0 или 1) был прочитан из защищённой области памяти Ap.

Повторяя этот алгоритм для других битов значения V(Ap), можно получить всё содержимое защищённой области памяти целиком. Перед этой атакой уязвимы все существующие ОС на большинстве современных процессоров, так как модель виртуального адресного пространства, содержащего память ядра ОС в адресном пространстве каждого процесса, является основной. Уязвимы так же и облачные системы и системы виртуализации. Существует

надежный программный способ борьбы с атакой, при котором в таблице страниц пользовательских процессов не отображаются страницы памяти ядра ОС (за исключением небольшого количества служебных областей памяти ядра), технология Kernel page-table isolation (KPTI). При этом несколько замедляются вызовы со сменой уровня привилегий (в частности, системные вызовы), так как им приходится дополнительно переключаться на другую таблицу страниц, описывающую всю память ядра ОС.

- Корпорация Microsoft выпустила экстренное обновление Windows 10 для предотвращения атаки 3 января 2018, ожидается выход аналогичных патчей для других поддерживаемых версий ОС Windows в следующий Вторник Патчей.
- Разработчики ядра Linux предложили набор патчей по имени Kernel page-table isolation (KPTI, рабочие названия KAISER, UASS, FUCKWIT), который вошёл в ядро версии 4.15 в начале 2018 года и портирован в ядро версии 4.14.11.
- Ядро macOS получило исправление в версии 10.13.2.

В некоторых случаях исправление может снижать производительность ряда функций, например, приложений, очень часто выполняющих системные вызовы. Многие издания и эксперты проводили исследования производительности и получали довольно разнообразные результаты. Например, таблица результатов из статьи iChip.

Потери производительности после январского обновления (в процентах)

	Gigabyte Sabre 17 Intel Core i7-7700HQ	Acer Nitro 5 Spin Intel Core i5-8250U
PC Mark 7		
PC Mark 7	-0.7	-0.6
Graphics DX9	-1.2	1.5
Image manipulation	-5.4	-5.4

Web browsing	-3.8	-2.6

В PC Mark 8 получены схожие результаты. Однако в серверных приложениях ситуация значительно хуже. На более низком уровне потери производительности оказались значительно более серьезными. Прежде всего, пострадал доступ к жесткому диску и твердотельному накопителю. Для операционной системы и процессора каждый доступ к накопителю большой емкости означает критический переход с пользовательского контекста на системный, а обновление его замедляет. Производительность высокоскоростных твердотельных накопителей, работающих по протоколу NVMe, которые могут выполнять до 200 000 операций ввода-вывода в секунду, в таком случае может упасть примерно в два раза. Существенно выросла нагрузка на процессоры серверов баз данных.

Частично обе уязвимости можно устранить отключением технологии многопоточности:

- Hyperthreading для Intel
- SMT для AMD

В декабре 2018 года обнаружена ещё одна уязвимость — **Spoiler**. Уязвимость Spoiler связана со спекулятивным исполнением команд – техникой, которая используется для повышения производительности процессоров. Тем не менее, Spoiler затрагивает другую область процессора под названием буфер изменения порядка обращения (Memory Order Buffer). Данная область отвечает за управление операциями в памяти и тесно связана с кэшем.

Эксплуатация утечки возможна с помощью ограниченного набора инструкций, которые являются видимыми во всех поколениях процессоров Intel, начиная от Intel Core 1-го поколения. Атаку можно провести даже в виртуальных машинах и изолированных средах (в браузере через JavScript к примеру). Полное устранение

уязвимости средствами только программных заплаток не представляется возможным.
В процессорах AMD механизмы управления памятью используют другие алгоритмы и по заверениям представителей AMD не подвержены этому типу атак (на март 2019 года).
В мае 2019 года обнаружены новые уязвимости

- RIDL – утечка через порты и буферы заполнения
- Fallout – утечка из буфера промежуточного хранения CPU
- ZombieLoad – утечка из общего кэш ядер

А в августе 2019 **SWAPGSattack**, которая позволяет алфавитным перебором и просмотром данных ядра ОС узнать пароль пользователя.
Не смотря на собственные названия, эти уязвимости основаны на тех же принципах, что и ранее найденные: спекулятивное исполнение команд. Рекомендации intel – обновить процессоры не ниже 8 серии intel Core или intel Xeon 2; рекомендации независимых экспертов — отключение hyperthreading + программные патчи.
ZombieLoad v.2 обнаружена в ноябре 2019. Эта уязвимость с официально присвоенным индексом CVE-2019-11135 эксплуатирует операцию асинхронного прерывания в механизме Transactional Synchronization Extensions. В Intel назвали уязвимость TAA (TSX Asynchronous Abort). Злоумышленник, используя TAA и вредоносный код, запускаемый на атакуемом процессоре, вызывает конфликт между операциями чтения в процессоре. Конфликт ведёт к утечке данных о том, что обрабатывается процессором. Эта уязвимость затрагивает все процессоры компании с технологией TSX, а внедрена она была с 2013 года и, как видим, даже перекочевала в неизменном (уязвимом) виде в самые современные процессоры Intel.
По словам Intel, уязвимость Zombieload v2, во-первых, уже закрыта микрокодом и, во-вторых, трудно реализуема на практике. Компания не считает эту атаку чем-то опасным, хотя рекомендует

обновить микрокод всем, кто оперирует чувствительными к утечкам данными. Но даже запуск на процессоре вредоносного кода с использованием Zombieload v2 не даёт никаких гарантий кражи чувствительных данных, заявляют в Intel. Злоумышленник не сможет контролировать утечки и извлекать то, что ему нужно. В крайнем случае, предлагают в Intel, просто отключите поддержку процессором инструкций TSX. Это снизит производительность при обработке больших массивов данных, но позволит спать спокойно.