

Halbleitertechnologie von A bis Z

Grundlagen

Bulk vs. SOI	3
<i>Einordnung</i>	3
<i>Aufbau im Vergleich</i>	3
<i>Parasitäre Effekte und Isolation</i>	4
<i>Fertigung und Kosten</i>	4
<i>Vergleichstabelle</i>	5

Bulk vs. SOI

Einordnung

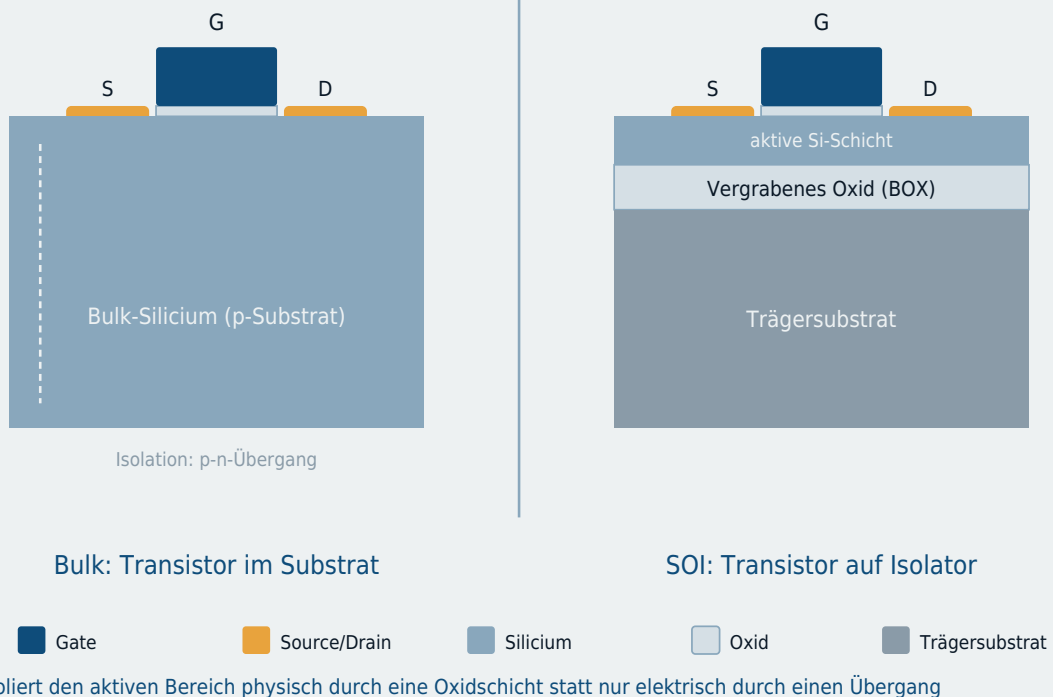
Bulk und SOI (Silicon-on-Insulator) sind keine unterschiedlichen Transistortypen, sondern zwei unterschiedliche Wafer-Technologien – die Frage ist nicht, *welcher* MOSFET gebaut wird, sondern *worauf* er sitzt. Beide Ansätze lassen sich mit denselben Grundprinzipien aus dem [MOSFET-Kapitel](#) kombinieren; der Unterschied liegt ausschließlich im Substrat darunter und wie der aktive Bereich davon elektrisch getrennt ist.

Aufbau im Vergleich

Bei der klassischen Bulk-Technologie sitzt der Transistor direkt in einem massiven, homogenen Siliciumwafer. Die elektrische Trennung von Nachbarbauteilen und vom restlichen Substrat erfolgt über dotierte Wannen und die dabei entstehenden gesperrten [p-n-Übergänge](#) – eine elektrische, aber keine physische Isolation.

SOI geht einen anderen Weg: Unter der dünnen, aktiven Siliciumschicht, in der die Transistoren gefertigt werden, liegt eine vergrabene Oxidschicht (BOX, Buried Oxide) aus Siliciumdioxid, darunter erst das eigentliche Trägersubstrat. Diese Oxidschicht isoliert den aktiven Bereich physisch und vollständig vom Substrat – nicht nur elektrisch über eine Sperrschicht, sondern durch einen echten Isolator.

Bulk vs. SOI – Querschnitt im Vergleich



Parasitäre Effekte und Isolation

Diese physische Isolation hat direkte Folgen: Bei Bulk-Bauelementen existiert immer eine parasitäre Kapazität zwischen dem aktiven Gebiet und dem Substrat, die Schaltgeschwindigkeit und Leistungsaufnahme beeinträchtigt. Zudem können benachbarte n- und p-Wannen zusammen mit dem Substrat eine parasitäre Thyristorstruktur bilden, die unter bestimmten Bedingungen zündet (Latch-up) und die Schaltung funktionsunfähig macht oder sogar zerstört.

Bei SOI entfällt die Substratkapazität praktisch vollständig, da die vergrabene Oxidschicht dazwischenliegt – das ermöglicht höhere Schaltgeschwindigkeiten bei geringerer Verlustleistung. Latch-up ist strukturell ausgeschlossen, da kein durchgehender Halbleiterpfad zum Substrat existiert. Zusätzlich verbessert die dünne, vollständig isolierte aktive Schicht das Kurzkanalverhalten: Das elektrische Feld vom Drain kann sich nicht mehr so leicht durch das Substrat zum Kanal ausbreiten, was unerwünschte Kurzkanaleffekte wie DIBL reduziert – ein Vorteil, den moderne Prozesse ähnlich wie beim [FinFET](#) zur weiteren Skalierung nutzen.

Fertigung und Kosten

Bulk-Wafer sind Standardware: großflächig verfügbar, seit Jahrzehnten auf hohe Ausbeute optimiert und vergleichsweise günstig. SOI-Wafer benötigen dagegen einen zusätzlichen Fertigungsschritt, um die vergrabene Oxidschicht überhaupt erst zu erzeugen – meist durch Wafer-Bonding (zwei oxidierte Wafer werden verbunden, einer anschließend auf die gewünschte Dicke zurückgeschliffen) oder durch SIMOX (Sauerstoff-Ionenimplantation mit anschließender Temperung, die eine vergrabene Oxidschicht direkt im Wafer entstehen lässt). Beide Verfahren machen SOI-Wafer deutlich teurer als Bulk-Material.

Man unterscheidet außerdem zwischen PD-SOI (Partially Depleted) und FD-SOI (Fully Depleted): Bei PD-SOI ist die aktive Siliciumschicht dick genug, dass ein neutraler, „potentialfreier“ Bereich im Kanal verbleibt – ähnlich wie bei Bulk, aber mit reduzierter Substratkapazität. Bei FD-SOI ist die Schicht so dünn, dass sie unter Betriebsspannung vollständig verarmt (kein neutrales Gebiet mehr übrig bleibt), was das Kurzkanalverhalten nochmals verbessert und heute vor allem in energieeffizienten und Mixed-Signal-Anwendungen genutzt wird.

Vergleichstabelle

Merkmal	Bulk	SOI
Isolation zum Substrat	elektrisch (p-n-Übergang)	physisch (Oxidschicht, BOX)
Parasitäre Kapazität	vorhanden	stark reduziert
Latch-up-Risiko	vorhanden	strukturell ausgeschlossen
Kurzkanalverhalten	Standard	verbessert
Kosten	niedrig	deutlich höher
Typische Anwendung	Massenmarkt-Logik	Hochleistungs-/Low-Power-/RF-Anwendungen