UNIVERSITETET I SØRØST NORGE

DATASYSTEMDESIGN

EN_DAT2000-1 18V

# TPU og maskinlæring

*Forfatter*
Magnus HAUKEBØE

*Forfatter*
Sigurd HOLMEN

# Innholdsfortegnelse

# 1 Innledning

CPU-er har med årene blitt raskere og økt antall beregninger den kan gjøre i løpet av en tidsperiode. Dette er på grunn av prosessor utviklere har vært i stand til å minske transistoren.

# 2 TPU

Nevrale nettverk og «deep learning» får stadig mer popularitet, og det oppdages stadig nye bruksområder. Til et nevralt nettverk kreves det en god del prosesseringskraft, og når prosessorutviklingen ikke er like rask som før, må en se på nye løsninger. En har begynt å se nærmere på hardware som er bygget for spesielle oppgaver, såkalt domenespesifikk hardware.

IT giganten Google har i mange av sine applikasjoner behov for maskin læring. Alt i fra tale- og bildegjenkjenning, til å slå eksperter i komplekse spill som Go [**look_at_TPU**]. I 2006 gjorde de en intern undersøkelse på behovet deres. Den gangen fant de ut at det var tilgjengelig databehandling i datasentrene, og derfor ikke trengte noe mer. I følge publikasjonen, var det i 2013 mer bruk av talesøk som førte til at Google måtte doble datasentrene for å møte behovet. I stedet for å bygge flere, valgte de å designe en ny hardware akselerator som kan gjøre denne jobben raskere og med et mindre energiforbruk enn tradisjonelle CPU-er og GPU-er.

I en publikasjon av Google [**tpu_main**] tar de for seg de ulike elementene hos den første versjonen av TPU-en. Deres største behov i 2013 var å kjøre hele inferens-modeller raskt og energieffektivt. Ved å bruke matrise regning kan en effektivisere prosessen, som kan bli utført på tidligere teknologi, men som TPU-en har blitt spesifikt designet til å gjøre jobben bedre.

Denne versjonen av TPU-en er bygget med transistorer på 28nm i størrelse. Den kjører 700 MHz og har et effektforbruk på 40 W [**tpu_main**]. Den er laget med en versjon 3 PCIe med *16 datarate som tillot å sette ASIC-chipen rett inn i eksisterende servere som allerede bruker dette grensesnittet. I artikkelen sier de at de får en 12.5GB/s effektiv båndbredde mellom TPUen og prosessoren som styrer.

## 2.1 Unike trekk

Men så er spørsmålet, hvis vi ikke skal bruke transistorene på å lage en generell prosesserings enhet, hva skal vi bruke dem på? Her har Google tenkt igjennom hva som behøves og ikke, og laget hardware som har en bedre ytelse på operasjoner som et nevralt nett trenger. Om en tar en titt på figur 1, kan en se at 24% av transistorene går til en spesiell hardware bit kalt 'Matrix Multiply Unit'. Dette er en stor del av TPU-en og utfører kjerne funksjonen dens.

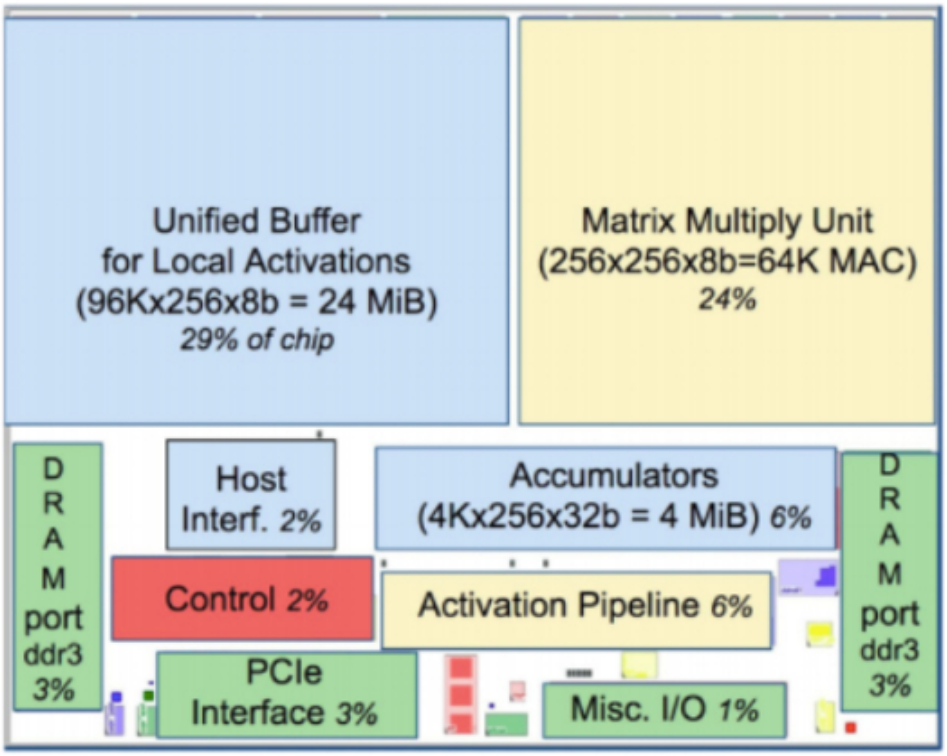


Figure 1: Transistorfordeling

RISC-prosessoren har enkle instruksjoner som i sammen kan gjøre det meste en ønsker. En instruksjon kan for eksempel være å legge sammen eller multiplisere, men det kreves flere klokkeperioder for å gjennomføre mer kompliserte funksjoner(satt sammen av flere instrukser). Noen CPU-er og GPU-er implementerer 'vector processing' [**look_at_TPU**] som innebærer at de kan utføre den samme instruksjonen for flere data(SIMD) på en periode.

Det TPU-en gjør annerledes er å innføre noe som kalles systolic array. Denne fungerer ved at verdier som kommer ut av en ALU blir sendt videre til neste ALU i rekken og utfører sammen en matrise beregning. Slik som i figur 2 er et slikt system mer effektivt enn en tradisjonell overføring mellom register og ALU som finnes i en CPU. Antall transistorer som trengs for å bygge en 8-bit ALU er også mindre enn 32- og 64-bit, som gjør at det er plass til flere på samme areal. Til sammen har TPU-en 256 * 256 ALU-er i sin MMU[**look_at_TPU**]. Dette tilsvarer 65,536 ALU-er og når klokkefrekvensen er på 700 MHz blir det totalt $4,6 * 10^{13}$ multiplikasjon/add operasjoner per sekund.

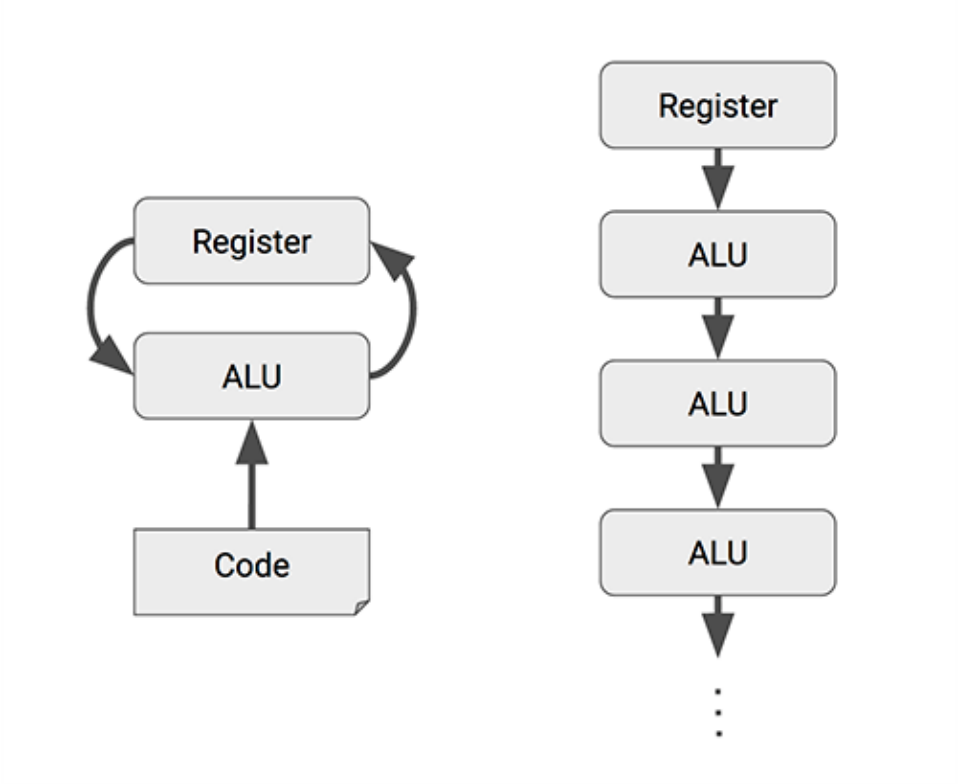


Figure 2: Konsept til Matrix multiply unit

Si at du skal regne ut hvor mange biler som kjører langs en motorvei, og du vil bare måle hvor mye kø det er når du skal lage en rute. Det er da ikke nødvendig med å vite akkurat hvor mange biler det er, bare et estimat om det er mange eller få. Samme kan sies i et nevralt nettverk. Om vi tenker oss nøyaktigheten til 32-bits flyttall trenger et typisk nevralt nettverk vanligvis ikke denne nøyaktigheten for hver

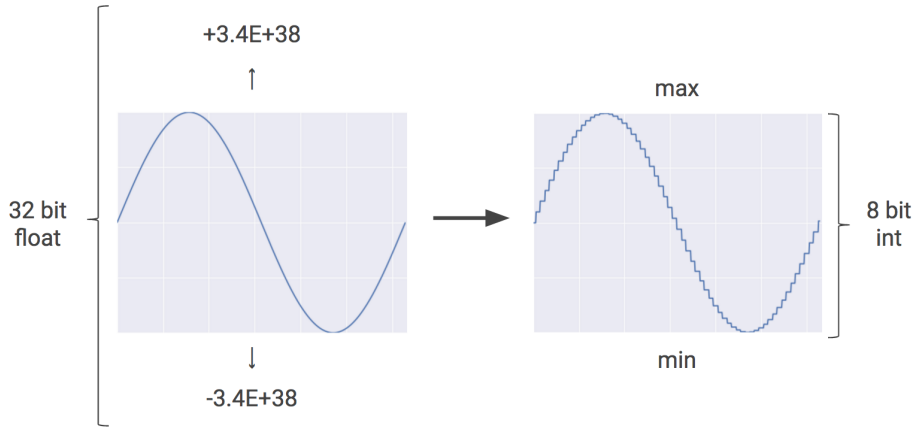


Figure 3: Kan oppnå en viss nøyaktighet selv med færre bits

node. I figur 3 ser vi at 8 bit (0 til 256) kan gi oss en tilnærmet kurve som dekker behovet. Ved å bruke quantization kan en finne 8-bits tallverdier mellom minimum og maksimum. [**look_at_TPU**].

Brikken er også blitt laget så enkelt som mulig, og ungår mange nødvendigheter som finnes i dagens CPU-er som branch prediction, out-of-order execution, cacher, osv. Dette reduserer hvor mange transistorer som må brukes, og i figur 1 ser en at kun 2% av hele chipen blir brukt til kontrollflyten [**tpu_main**]. Vekter blir for eksempel lagret i en read-only DRAM, som fjerner problemet ved parallell programmering (siden ingen kan skrive over minnet, har alle samme informasjonen).

TPU-en ble designet til å være en koprosessor til en CPU, og ved å bruke PCIe I/O busser, tillot det å putte den nye ASCI chipen rett inn i servere, slik som en kan gjøre med GPU-er. Den mottar TPU-instruksjoner ifra hosten sin, og bruker CISC (Complex Instruction Set Computer) prinsippet, for å kutte ned på antall instruksjoner som må bli overført. Instruksjonssettet inneholder rundt et dusin instruksjoner, der den gjennomsnittlige CPI-en er rundt 10-20. Selv med en 4-stegs pipeline, så kan samme instruksjon bli kjørt over tusen ganger igjennom samme område, i motsetning til RISC-prosessoren som kjører igjennom en pipeline avdeling per klokkesyklus. I blokkdiagramet i figur 4 ser vi flyten inne i chipen. MMU-en får data ifra Weight FIFO-en og fra et unified buffer. Resultatet fra utregningene blir den sendt og lagret i accumulatorerer.

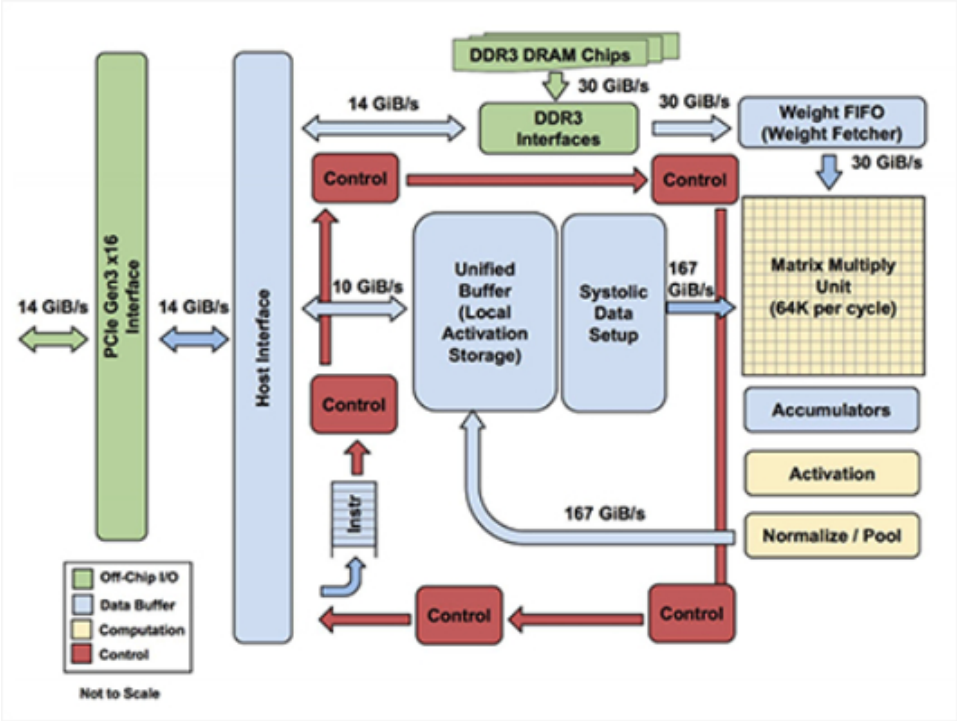


Figure 4: Blokkdiakgram for TPU-en

# 3 Nyere Versjoner

Siden den første versjonen av TPU-en kom ut i 2015 har det kommet nyere utgaver [**tpu_video**]. Der den første har 92 teraops og kun kunne brukes til inferens, har det kommet en cloud TPU (som vi hadde tenkt til å bruke i den praktiske delen av prosjektet) som har dobbelt så mange operasjoner per sekund og kunne brukes til trening i tillegg. Den har også støtte for flyttall. Det er i tillegg mulig å koble flere Cloud TPUer til et cluster(kalt TPU Pod) som har opp til 11.5 petaflops. I 2018 kommer versjon 3 av TPU Pod som klarer over 100 petaflops.

# 4 Benchmarks

Når en skal utføre benchmark på nevrale nettverk er det flere ting en kan måle. Det kan være hvor lang tid det tar å nå en viss nøyaktighet, hvor mye det koster å kjøre algoritmen i skyen (server leie). Det er også viktig at det er samme datasettet som blir testet, og i en benchmark[**benchmark**] hostet hos Standford omhandler Image Classification som benytter seg av open-source bilder fra image-net [**image-net**].

## 4.1 Treningstid

I ulike kategorier blir ulike hardwareoppsett, modeller og rammeverk målt opp mot hverandre, og blir her målt i tiden treningen bruker for å nå en 93% sikkerhet. På de tre første plassene på toppen ligger google sin TPU-pod oppdelt i 1/2(0:30:43), 1/4(1:06:32) og 1/16(1:58:24). Neste på listen ligger på 2:57:28 som kjører et oppsett ifra Amazone sin skytjeneste.

## 4.2 Treneingskostnad

Denne kategorien går ut på å oppnå den minste prisen for å trene opp til 93%. Dessverre er det ikke oppgitt hvor mye topp 3 koster å trene. Amazone sin tjeneste kostet 72 USD, og første prisgitte TPU-innslaget koster 49 USD, men tok 7:28:30. Her må en vurdere hva som er viktigst av pris og ytelse.