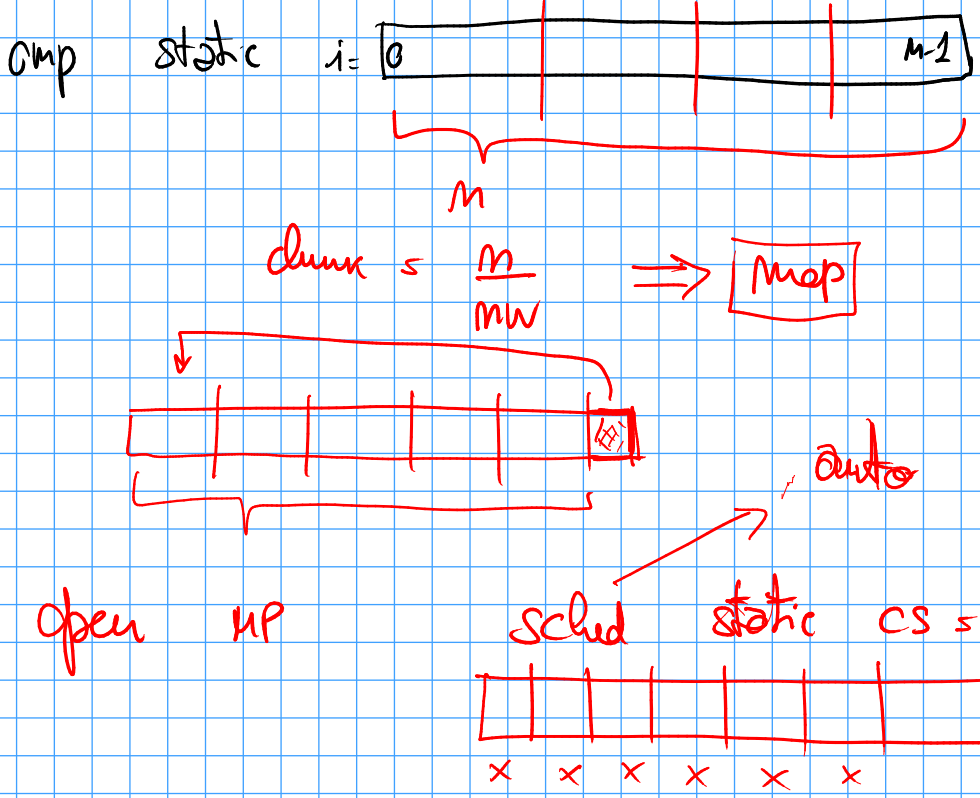
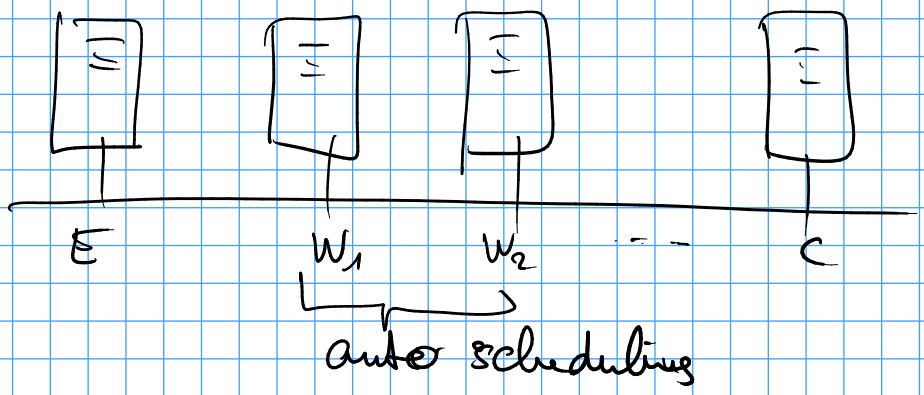
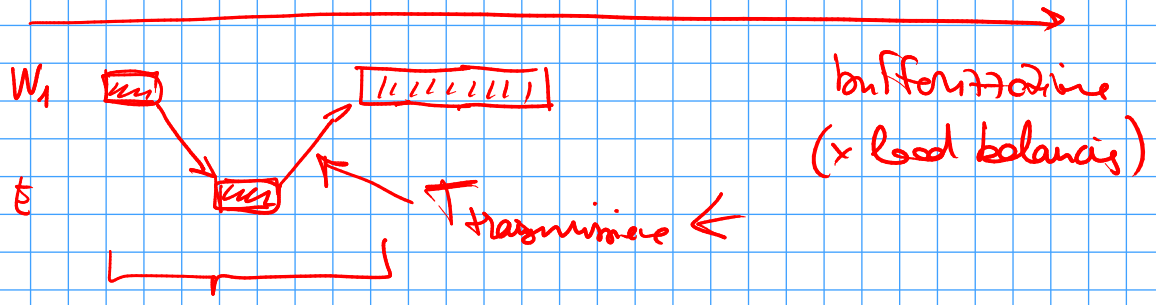
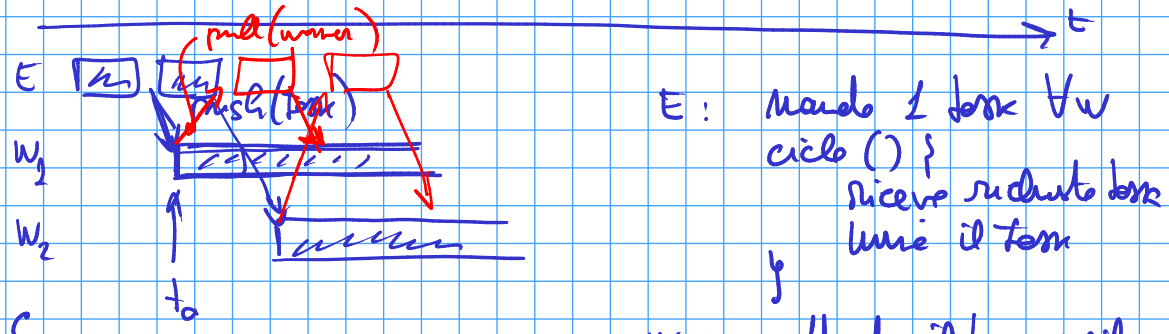

omp static i=
0
n-1
n
chunk = n/nw
map
open MP
sched static cs = X
auto

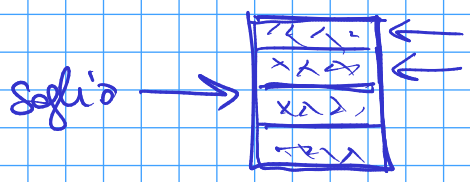


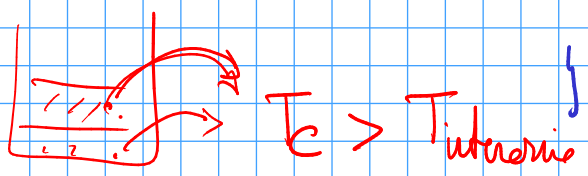




E: Manda 1 task $\forall w$
ciclo () {
riceve richiesta task
invia il task
}

W: attende il task inviato
ciclo () {
Calcolo n. disp
(dei buffer)
Se nei buffer
ho n < soglia task
da calcolare
→ manda una richiesta
riceve il numero task
}





SkePU — Map — Reduce/Scan

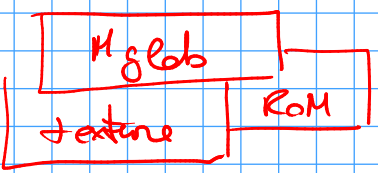
↳ target hw: CPU multicore — OpenCL, openMP ×
GPU — CUDA (nVidia) ×
OpenCL ×

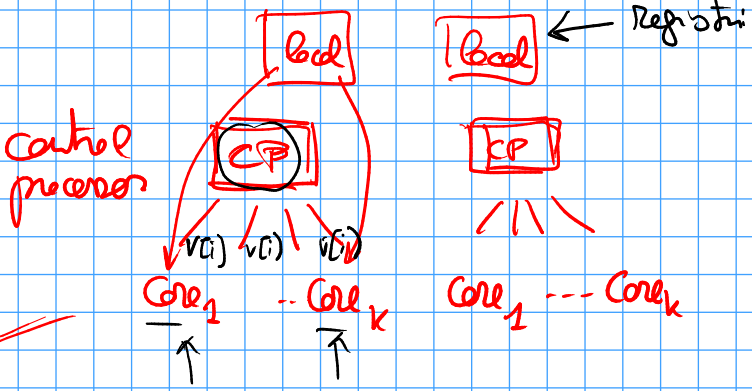
SKEPU_OMP
#define SKEPU_CUDA

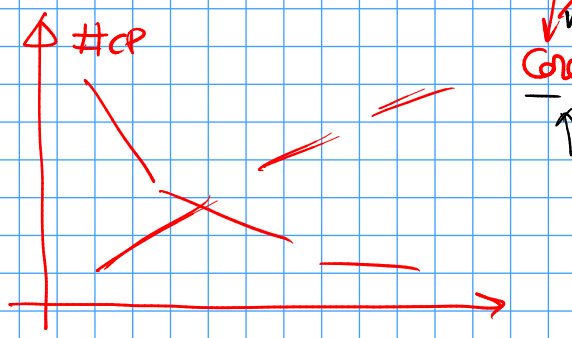
GP-GPU

↳ schede PCIe
↳ # core

→ griglia
→ block block
→ {{{{}}}}







SW

```
__kernel pippo(float *v, int n)
{
    int i = ....
    v[i] = f(v[i]);
}
```

↑ ↑ ↑

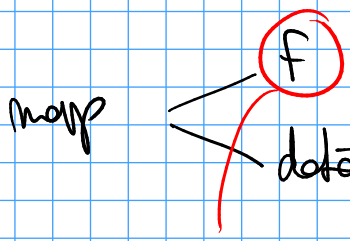
<<< dim >>> pippo(...)

skepu :: vector< >

std :: vector < >

collezioni

$y = map(f)(x)$



UNARY_FUNC (name, ~~type~~ param, name param,
Corpo fun)