→ Revise multiplication algos from DD!!

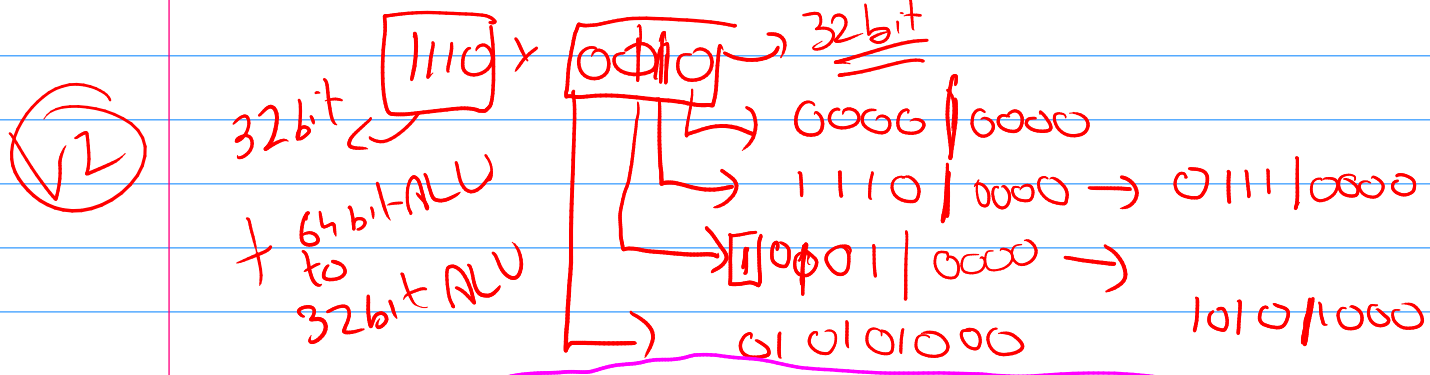
Shift add & Add shift

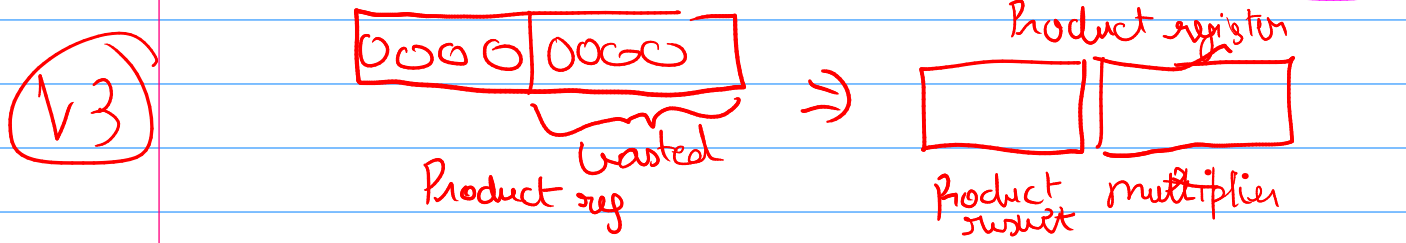
Problem for 64 bit registers,

multiplicand is 32 bits long

32 bits wasted

⇒ Intuition, instead of shifting multiplicand to left, shift product to right



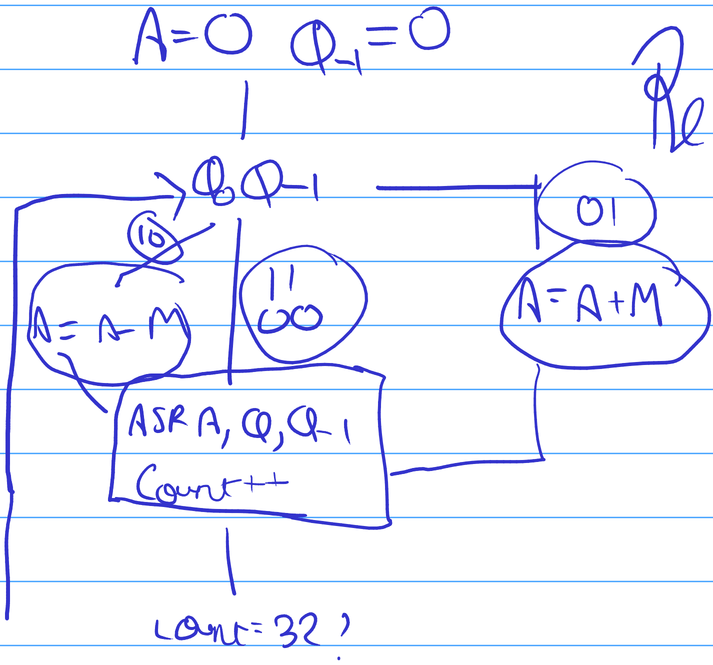


⇒ Same a multiplier

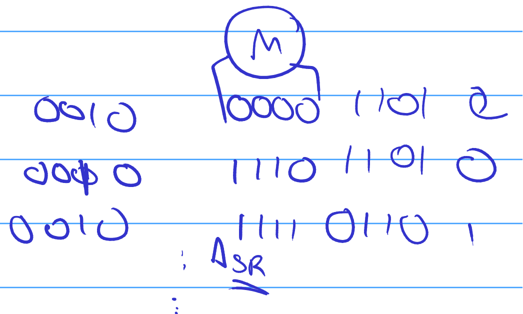
33% improvement in clock cycles

Signed integers

Booth's Algorithm



Revise



MIPS Notes

- MIPS provides two 32-bit registers `Hi` and `Lo` to hold a 64-bit product
- `mult`, `multu` (unsigned) put the product of two 32-bit register operands into `Hi` and `Lo`: overflow is ignored by MIPS but can be detected by programmer by examining contents of `Hi`
- `mflo`, `mfhi` moves content of `Hi` or `Lo` to a general-purpose register
- Pseudo-instructions `mul` (without overflow), `mulo` (with overflow), `mulou` (unsigned with overflow) take three 32-bit register operands, putting the product of two registers into the third

Resp time = tot exec time (runtime) + waiting time

Either or both decreases for resp ↑

throughput : no. of Jobs completed / time

throughput ↑ : parallelized or improve system resources

Individual time for job remains same!!

if throughput ↑

waiting time ↓ (with more resources)

∴ resp time ↑

if speed ↑ resp time ↑ throughput ↑

BUT If throughput ↑ execution time need not improve if we consider only execution time