* 1. A page table is a map between virtual memory addresses and physical memory addresses. It is stored in memory with a CPU register pointing to it. It is initialised to all zeros.
  2. One-level page tables would be large with much of the space unused. Multilevel page tables are represented as trees which is more efficient.
  3. The principles of locality also apply to page tables, meaning that nearby pages are likely to be accessed around the same time. The TLB is a cache of PTEs which bypasses the need perform two separate address lookups (one to the PTE and the other to the physical memory address).
  4. Environment calls are used to jump to privileged code and change privilege mode. Interrupts are caused by external factors rather than instruction execution. Exceptions are caused by something going wrong in the instruction stream.
  5. A register file is a bank of general-purpose registers which can be written to and read from by instructions. An accumulator is a special-purpose register which holds intermediate values from an ALU computation. Having an accumulator makes the ISA simpler because the arithmetic instructions don’t need to specify which register they will use for storing intermediate values.
  6. CISC has variable-length instructions which save on storage space but are complex to decode. CISC also has many special-purpose registers as opposed to RISC whose registers are largely orthogonal.
  7. The Cray-1 had a “minicomputer” which manages the “supercomputer”. This is similar to how modern devices have a smaller core handling low-energy background tasks while the phone is idle.
     + 1. Machine
          1. A tree-shaped multi-level page table
          2. The page table could be small and each page’s index in the page table could be given by the last few bits. Collisions would hopefully be rare and could be resolved using a linked list.
       2. A superpage is much larger than a normal page. Its lookups only require one level of indirection as opposed to normal pages which require two.
       3. The TLB is accessed before the L1 cache
       4. We use a TLB because page lookups require two memory accesses, but page accesses have good locality so nearby pages can be cached to reduce access time
       5. TLB entries often include the ASID so that the TLB can hold translations for different virtual address spaces.
       6. These capabilities include dynamic bounds checking and pointer integrity checks. This means that code is less vulnerable to exploits such as buffer overflows etc.