Static and Dynamic Relocation

Introduction
This shows the basic hardware instruction cycle for a machine that uses static relocation and for one that uses dynamic relocation.

Static Relocation
Static relocation refers to address transformations being done before execution of a program begins. A typical hardware instruction cycle looks like this:

```plaintext
loop
    w := M[instr_ctr]; (* fetch instruction *)
    oc := Opcode(w);
    adr := Address(w);
    instr_ctr := instr_ctr + 1;
    case oc of
    1:    reg := reg+M[adr]; (* add *)
    2:    M[adr] := reg; (* store *)
    3:    instr ctr := adr; (* branch *)
    ...
end
end (* loop *)
```

Dynamic Relocation
Dynamic relocation refers to address transformations being done during execution of a program. In what follows, the function NL_map (for Name Location map) maps the relocatable (virtual) address va given in the program into the real (physical) storage address pa:

```plaintext
pa := NL_map(va)
```

So, a typical hardware instruction cycle looks like this:

```plaintext
loop
    w := M[NL_map(instr_ctr)]; (* fetch instruction *)
    oc := Opcode(w);
    adr := Address(w);
    instr ctr := instr ctr + 1;
    case oc of
    1:    reg := reg+M[NL_map(adr)];(* add *)
    2:    M[NL_map(adr)] := reg; (* store *)
    3:    instr ctr := NL_map(adr);(* branch *)
    ...
end
end (* loop *)
```