The Theory of Computation
Bernard M.E. Moret
Addison-Wesley-Longman, 1998

Short Solution
Chapter 4, Problem 13

This solution is not completely formal: we do not give a proof of equivalence of the two formulations, just the construction. Yet, the construction is clear enough that the proof is clearly just a matter of routine. Thus, this solution is sufficient for communication in class or among people acquainted with the field—but it lacks the rigor needed for a publication.

We prove the slightly stronger result that a 2-register machine can simulate a k-register machine.

Note that any general RAM has some fixed number of registers (all those named in the program). We encode all of the registers of the general RAM into the first register of our 2-register machine by using prime encoding: if the general RAM uses k registers and their current values are r_i, $1 \leq i \leq k$, the first register or our machine will contain the value $\prod_{i=1}^{k} p_i^{r_i}$, where p_i is the ith prime. The fundamental theorem of arithmetic tells us that such an encoding is one-to-one. The second register of our machine will be used for temporaries. To simulate an increment of register i, our machine simply multiplies its first register by p_i; it can do that by first setting the second register to zero, then entering a loop that carries out p_i increments of the second register (something that can be done by an *ad hoc* routine, since p_i is a known constant) for one decrement of the first, until the first register is zero, then entering another loop that copies the result from the second back into the first register. To simulate a decrement of register i, our machine divides its first register by p_i, which proceeds in the same spirit as the multiplication, except that we carry out p_i decrements of the first register for each increment of the second. (Note that the last routine can be modified to test for the presence of a remainder, indicating that the register was 0 before the decrement and should remain unchanged; we can then invert the operation completely to recover the original value.) To test register i for zero, we carry out the division by p_i, test whether the remainder is non-zero (indicating there was no factor of p_i in the encoding and hence that we had $r_i = 0$), then remultiply by p_i in order to maintain the original value. Jumps themselves are of course no problem. Thus we have a valid simulation of any given k-register RAM on a 2-register RAM, so that 2-register RAMs (and thus also 3-register RAMs) are universal models of computation.