The address pins

The Z80 provides a 16-bit data bus to the outside world. This is a tri-state bus. It can output an address or sit in a high-impedance idle state. This idle state is important because it allows other devices to use the bus.

The address bus is fed from the registers, allowing the PC and R registers direct access. The other registers can access the address bus, but need to pass through the gate separating the two halves of the register file.

Like the data pins, the address pins have large transistors driving each pin. Also like the data pins, each address pin has a latch at the pin.

Four control lines form a bus just inside the address pins. From inside to outside, they are:
enable_addr/ (A): This control line goes low to drive the latched value to the address pin. When this control line is high, the pin is inactive, in a high-impedance state.

Clock: The signal from the address bus must pass through a pass transistors controlled by the clock.

Latch_addr (D): When this control line is high, the address latch holds its value. When this control line is low, the address latch can receive a new value.

abus_to_addr (E): When this control line is high, and the clock is high, the value from the address bus is loaded into the latch.

Note that the values on the address bus are inverted - the bus has a high voltage for a 0, and a low voltage for a 1. This value gets inverted before going to the address pin.

The enable_addr/ pin is low (check the simulator), except when the bus is taken over with a BUSREQ/BUSACK. The latch to manage this is next to the A0 pin. The logic to handle BUSREQ/BUSACK is closer to the BUSACK pin, not surprisingly.

The latch_addr control line is generated from the abus_to_addr control line. Specifically, the abus_to_addr control line is inverted to form the latch_addr control line. The inverter (superbuffer) to do this is near the A0 pin. Since the two control lines are inverses, this insures that the address pin latch will either be holding a value or receiving a new value.

The start of a memory address cycle is T1 or M1.T3). T1 is the start of every memory cycle. M1T3 is the refresh cycle. This signal, combined with mreq_latch, generates the abus_to_addr signal and latch_addr signals.

The latch_addr control line goes high in T1 or M1T3, latching the new address. The rest of the time, latch_addr is high, holding the address. enable_addr/ goes high.

The mreq_latch signal is used in a few cases. In a relative jump that is taken, it suppresses abus_to_addr during M3, preventing a new address from being loaded. Note that M3 in this case does not perform a memory access, but is just extra time needed to make the jump happen.

Likewise, ADD HL, DE has three M cycles (M1, M4, and M5), even though there is only one memory access (the op code). The address load is suppressed in M4 and M5.

A few other operations have “extra” M cycles that don't access memory: 16-bit ADC, SBC. This includes the index register ones.

RRD and RLD have M1 (prefix), M1 (opcode), M2 (read memory), M3, M4 (write memory). M3 is the “extra” M cycle that provides time to shift the bits around.

LDIR, CPIR, INIR, OTIR (and the corresponding -D- operations) have an extra memory cycle if the loop is taken. CPI, CPIR, CPD, and CPDR have an extra memory cycle to handle the 16-bit increment of HL and 16-bit decrement of BC.

The indexed instructions have an extra memory cycle to perform the index calculation. Typical cycles are M1 (prefix), M1 (opcode), M2 (displacement), M3 (index calculation), M4 (memory access).

These examples show that while M cycles are almost always tied to a memory access, there are a few cases where the T states aren't sufficient to perform a complex task, so an extra M cycle is added that doesn't perform a memory access.

The logic to compute the “skipped” M cycles is complex, with a lot of gates, combining a lot of PLA rows. The result is a set of instructions that skip addressing in M3, and a set of instructions that skip addressing in M4 or M5. These gates are in a wide variety of locations around the PLA, as they combine a lot of different PLA entries.

To summarize, normally the address is loaded into the latch during T1l/T1h of a M cycle (as well as T3l/T3h of M1 for refresh). The address is written to the bus constantly. Thus, the address won't necessarily be stable during the T1 cycle, but will be stable after that.