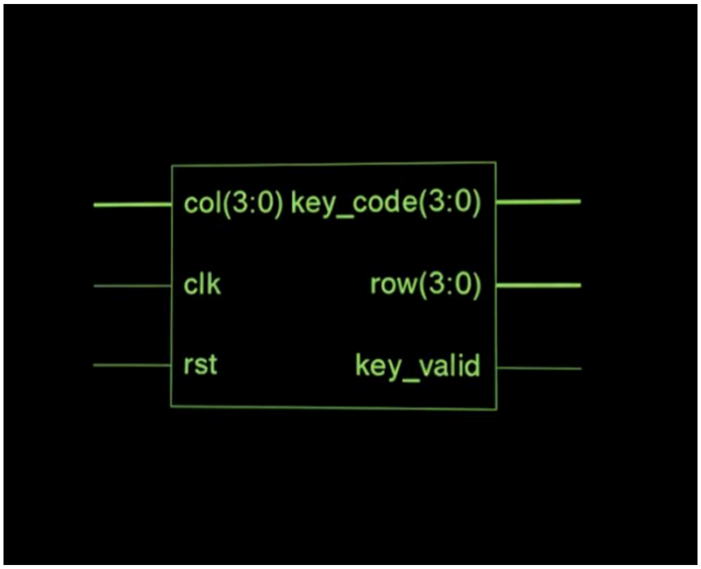
Keypad Code

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.NUMERIC_STD.ALL;

--------------------------------------------------------------------------------
-- ENTITY DECLARATION
--------------------------------------------------------------------------------
ENTITY keypad IS
   PORT (
      clk      : IN  STD_LOGIC;                        -- System clock
      rst      : IN  STD_LOGIC;                        -- Asynchronous reset
      Row      : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);     -- Keypad row outputs (active low)
      Col      : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);     -- Keypad column inputs (active low)
      Key_code : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);     -- 4-bit code of the pressed key
      Key_valid: OUT STD_LOGIC                         -- Signal to indicate a valid key press
   );
END keypad;

--------------------------------------------------------------------------------
-- ARCHITECTURE DEFINITION
--------------------------------------------------------------------------------
ARCHITECTURE Behavioral OF keypad IS

   -- State machine signal. The state determines which row to scan.
   SIGNAL state : INTEGER RANGE 0 TO 3 := 0;

BEGIN

   -- This process contains the main logic for the keypad scanner.
   -- It is sensitive to the clock and reset signals.
   PROCESS (clk, rst)
   BEGIN
      -- Asynchronous reset logic
      IF rst = '1' THEN
         Row       <= "1111";          -- Disable all rows (set high)
         Key_code  <= (OTHERS => '0'); -- Reset key code
         Key_valid <= '0';             -- Invalidate key
         state     <= 0;               -- Reset to the first state

      -- Logic executes on the rising edge of the clock
      ELSIF rising_edge(clk) THEN

         -- NOTE: Based on the provided code, 'key_valid' is set to '1' on a keypress
-- but is not automatically reset. An external module would need to
-- read the key and then handle the signal.

-- State machine for scanning rows
CASE state IS

  -- STATE 0: Scan Row 0
  WHEN 0 =>
    Row <= "1110"; -- Enable Row 0 (set low)
    -- Check each column for a key press
    IF col(0) = '0' THEN key_code <= "0000"; key_valid <= '1'; state <= 1; END IF;
    IF col(1) = '0' THEN key_code <= "0001"; key_valid <= '1'; state <= 1; END IF;
    IF col(2) = '0' THEN key_code <= "0010"; key_valid <= '1'; state <= 1; END IF;
    IF col(3) = '0' THEN key_code <= "0011"; key_valid <= '1'; state <= 1; END IF;

  -- STATE 1: Scan Row 1
  WHEN 1 =>
    Row <= "1101"; -- Enable Row 1
    IF col(0) = '0' THEN key_code <= "0100"; key_valid <= '1'; state <= 2; END IF;
    IF col(1) = '0' THEN key_code <= "0101"; key_valid <= '1'; state <= 2; END IF;
    IF col(2) = '0' THEN key_code <= "0110"; key_valid <= '1'; state <= 2; END IF;
    IF col(3) = '0' THEN key_code <= "0111"; key_valid <= '1'; state <= 2; END IF;

  -- STATE 2: Scan Row 2
  WHEN 2 =>
    Row <= "1011"; -- Enable Row 2
    IF col(0) = '0' THEN key_code <= "1000"; key_valid <= '1'; state <= 3; END IF;
    IF col(1) = '0' THEN key_code <= "1001"; key_valid <= '1'; state <= 3; END IF;
    IF col(2) = '0' THEN key_code <= "1010"; key_valid <= '1'; state <= 3; END IF;
    IF col(3) = '0' THEN key_code <= "1011"; key_valid <= '1'; state <= 3; END IF;

  -- STATE 3: Scan Row 3
  WHEN 3 =>
    Row <= "0111"; -- Enable Row 3
    IF col(0) = '0' THEN key_code <= "1100"; key_valid <= '1'; state <= 0; END IF;
    IF col(1) = '0' THEN key_code <= "1101"; key_valid <= '1'; state <= 0; END IF;
    IF col(2) = '0' THEN key_code <= "1110"; key_valid <= '1'; state <= 0; END IF;
    IF col(3) = '0' THEN key_code <= "1111"; key_valid <= '1'; state <= 0; END IF;

  -- Default case for safety
  WHEN OTHERS =>
    state <= 0; -- Reset state machine

END CASE;
```

```
    END IF;
  END PROCESS;

END Behavioral;
```

RTL Design


col(3:0)
key_code(3:0)
clk
row(3:0)
rst
key_valid