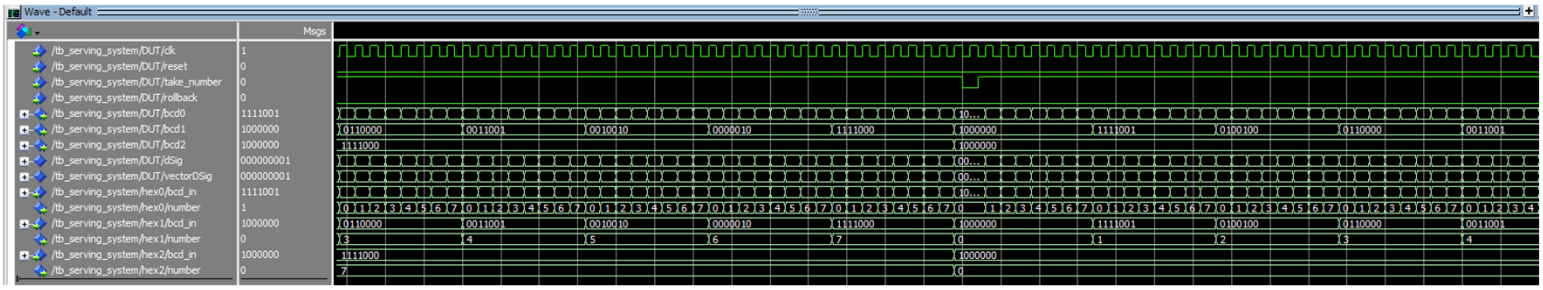
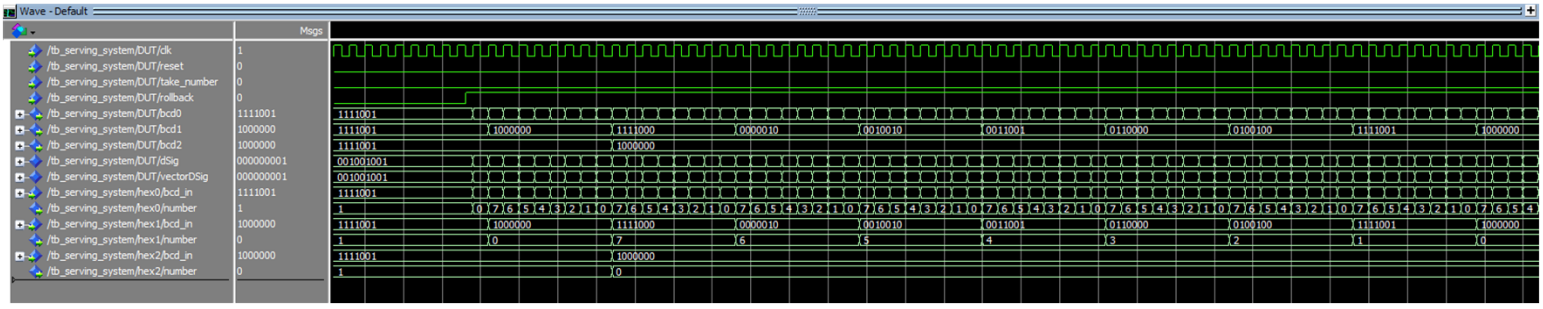
Bonus octal waveforms



Screenshot showing octal overflow from 777 to 000, along with other examples of incrementing in many aspects



Decrementing! All working as expected, even when underflowing. Not much to highlight here.

Bonus octal serving_system vhdl

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

ENTITY serving_system IS
    GENERIC (
        radix : INTEGER := 9;
        data_width : INTEGER := 4);
    PORT (
        clk, reset, take_number : IN STD_LOGIC;
        rollback : IN STD_LOGIC;
        bcd0, bcd1, bcd2 : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
END serving_system;

ARCHITECTURE structure OF serving_system IS

    --components
    COMPONENT SevenSeg IS
        PORT (
            D : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
            Y : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
    END COMPONENT;

    COMPONENT counter_chain IS
        GENERIC (
            radix : INTEGER := 9;
            data_width : INTEGER := 4);
        PORT (
            clk, reset, take_number : IN STD_LOGIC;
            rollback : IN STD_LOGIC;
            number : OUT UNSIGNED((3 * data_width) - 1 DOWNTO 0));
    END COMPONENT;

    --signals
    SIGNAL dSig : UNSIGNED((3 * data_width) - 1 DOWNTO 0);
    SIGNAL vectorDSig : STD_LOGIC_VECTOR ((3 * data_width) - 1 DOWNTO 0);

BEGIN

    vectorDSig <= STD_LOGIC_VECTOR(dSig);

    counter : counter_chain GENERIC MAP(radix, data_width)
    PORT MAP(clk, reset, take_number, rollback, dSig);

    seg0 : SevenSeg PORT MAP(D(3) => '0', D(2 downto 0) => vectorDSig((data_width - 1) DOWNTO
0), Y => bcd0);
    seg1 : SevenSeg PORT MAP(D(3) => '0', D(2 downto 0) => vectorDSig((2 * data_width - 1)
DOWNTO data_width), Y => bcd1);
    seg2 : SevenSeg PORT MAP(D(3) => '0', D(2 downto 0) => vectorDSig((3 * data_width - 1)
DOWNTO 2 * data_width), Y => bcd2);
END structure;
```

The main difference here is the modification of the seven-segment display port mapping, as I had to prepend a 0 to the beginning in order to make our 4-bit inputs work with a 3-bit number.

Bonus octal testbench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE IEEE.numeric_std.ALL;
--testbenches have empty entity sections
ENTITY tb_serving_system IS
END tb_serving_system;
ARCHITECTURE test OF tb_serving_system IS
component serving_system IS GENERIC (radix : INTEGER := 9; data_width : INTEGER := 4);
    PORT (clk, reset, take_number : IN STD_LOGIC; rollback : IN STD_LOGIC;
        bcd0, bcd1, bcd2 : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
END component;
COMPONENT hex_display IS
PORT (bcd_in : IN STD_LOGIC_VECTOR(6 DOWNTO 0); number : OUT STD.STANDARD.INTEGER);
END COMPONENT;
    --signals and constants
    CONSTANT data_width : INTEGER := 3;
    CONSTANT radix : INTEGER := 7;
    CONSTANT HALF_PERIOD : TIME := 10 ns;
    CONSTANT PERIOD : TIME := 20 ns;
    SIGNAL sigclk : STD_LOGIC := '1';
    SIGNAL sigreset, sigtake, sigrollback : STD_LOGIC;
    signal sigBCD0, sigBCD1, sigBCD2 : std_logic_vector (6 downto 0);
    SIGNAL sevenSegOut0 : INTEGER := 0;
    SIGNAL sevenSegOut1 : INTEGER := 0;
    SIGNAL sevenSegOut2 : INTEGER := 0;
BEGIN
    DUT : serving_system GENERIC MAP(7, 3)
    PORT MAP(sigclk, sigreset, sigtake, sigrollback, sigBCD0, sigBCD1, sigBCD2);
    hex0 : hex_display PORT MAP (sigBCD0, sevenSegOut0);
    hex1 : hex_display PORT MAP (sigBCD1, sevenSegOut1);
    hex2 : hex_display PORT MAP (sigBCD2, sevenSegOut2);
    sigclk <= NOT sigclk AFTER HALF_PERIOD;
    PROCESS IS
    BEGIN
        sigreset <= '1';
        sigtake <= '0';
        sigrollback <= '0';
        WAIT FOR HALF_PERIOD;
        sigreset <= '0';
        sigtake <= '1';
        WAIT FOR 8*8*8 * PERIOD;
        sigtake <= '0';
        WAIT FOR PERIOD;
        sigtake <= '1';
        WAIT FOR (1+8+8*8) * PERIOD;
        sigtake <= '0';
        WAIT FOR 10*PERIOD;
        sigrollback <= '1';
        WAIT FOR (1+8+8*8) * PERIOD;
        sigrollback <= '0';
        WAIT FOR PERIOD;
        sigrollback <= '1';
        WAIT FOR 2 * PERIOD;
        sigrollback <= '0';
        WAIT FOR PERIOD;
        sigtake <= '1';
        WAIT FOR PERIOD;
        sigtake <= '0';
        WAIT FOR PERIOD;
        WAIT;
    END PROCESS;
END test;
```

Differences here include the waiting periods, which are now based on powers of 8 instead of 10, and the fact that the DUT (serving_system) is instantiated with 7,3 as the radix and data width. Beyond that, all the files remained identical.