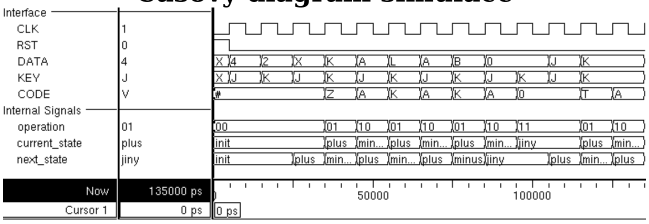
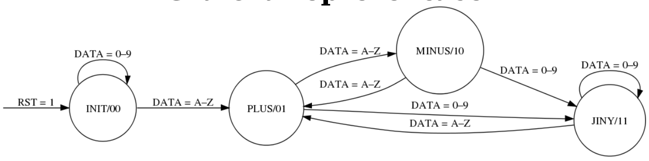
Jan Kaláb, xkalab00

zpráva:	4	2	X	K	A	L	A	B	0	0	J	K
klíč:	J	K	+J	−K	+J	−K	+J	−K	J	K	+J	−K
zašif.:	#	#	Z	A	K	A	K	A	0	0	T	A

Časový diagram simulace



Grafová reprezentace



Test bench

```
test : process
begin
    wait until clk'event AND clk = '1';
    rst <= '0';
    key  <= "01001010"; --J
    data <= "00110100"; --4
    wait until clk'event AND clk = '1';
    key  <= "01001011"; --K
    data <= "00110010"; --2
    wait until clk'event AND clk = '1';
    key  <= "01001010"; --J
    data <= "01011000"; --X
    wait until clk'event AND clk = '1';
    key  <= "01001011"; --K
    data <= "01001011"; --K
    wait until clk'event AND clk = '1';
    key  <= "01001010"; --J
    data <= "01000001"; --A
    wait until clk'event AND clk = '1';
    key  <= "01001011"; --K
    data <= "01001100"; --L
    wait until clk'event AND clk = '1';
    key  <= "01001010"; --J
    data <= "01000001"; --A
    wait until clk'event AND clk = '1';
    key  <= "01001011"; --K
    data <= "01000010"; --B
    wait until clk'event AND clk = '1';
    key  <= "01001010"; --J
    data <= "00110000"; --0
    wait until clk'event AND clk = '1';
    key  <= "01001011"; --K
    data <= "00110000"; --0
    wait until clk'event AND clk = '1';
    key  <= "01001010"; --J
    data <= "01001010"; --J
    wait until clk'event AND clk = '1';
    key  <= "01001011"; --K
    data <= "01001011"; --K
    wait until clk'event AND clk = '1';
end process;
```

FSM a ALU

```
sync_logic: process(CLK, RST) --RST
begin
    if (RST = '1') then
        current_state <= INIT;
    elsif (CLK'event AND CLK = '1') then
        current_state <= next_state;
end if;
end process sync_logic;

next_state_logic: process(current_state, DATA) --FSM
begin
    case (current_state) is
        when INIT =>
            if (DATA >= "01000001" AND DATA <= "01011010") then --AZ
                next_state <= PLUS;
            else
                next_state <= INIT;
            end if;
        when PLUS =>
            if (DATA >= "01000001" AND DATA <= "01011010") then --AZ
                next_state <= MINUS;
            else
                next_state <= jiny;
            end if;
        when MINUS =>
            if (DATA >= "01000001" AND DATA <= "01011010") then --AZ
                next_state <= PLUS;
            else
                next_state <= jiny;
            end if;
        when jiny =>
            if (DATA >= "01000001" AND DATA <= "01011010") then --AZ
                next_state <= PLUS;
            else
                next_state <= jiny;
            end if;
    end case;
end process next_state_logic;

vystup_logic: process(current_state, KEY) --vystupy
begin
    case (current_state) is
        when INIT =>
            operation <= "00";
        when PLUS =>
            operation <= "01";
        when MINUS =>
            operation <= "10";
        when jiny =>
            operation <= "11";
end case;
end process vystup_logic;

ALU: process(d_data, d_key, operation)
    variable temp: std_logic_vector(7 downto 0);
begin
    case (operation) is
        when "00" => --init
            code <= "00100011"; --#
        when "01" => --plus
            temp := d_data + (d_key - "01000000"); --@ (A-1)
            if (temp > "01011010") then --Z
                temp := "01011010"; --Z
            end if;
            code <= temp;
        when "10" => --minus
            temp := d_data - (d_key - "01000000"); --@ (A-1)
            if (temp < "01000001") then --A
                temp := "01000001"; --A
            end if;
            code <= temp;
        when "11" => --jiny
            code <= d_data;
        when others => null;
    end case;
end process;
```