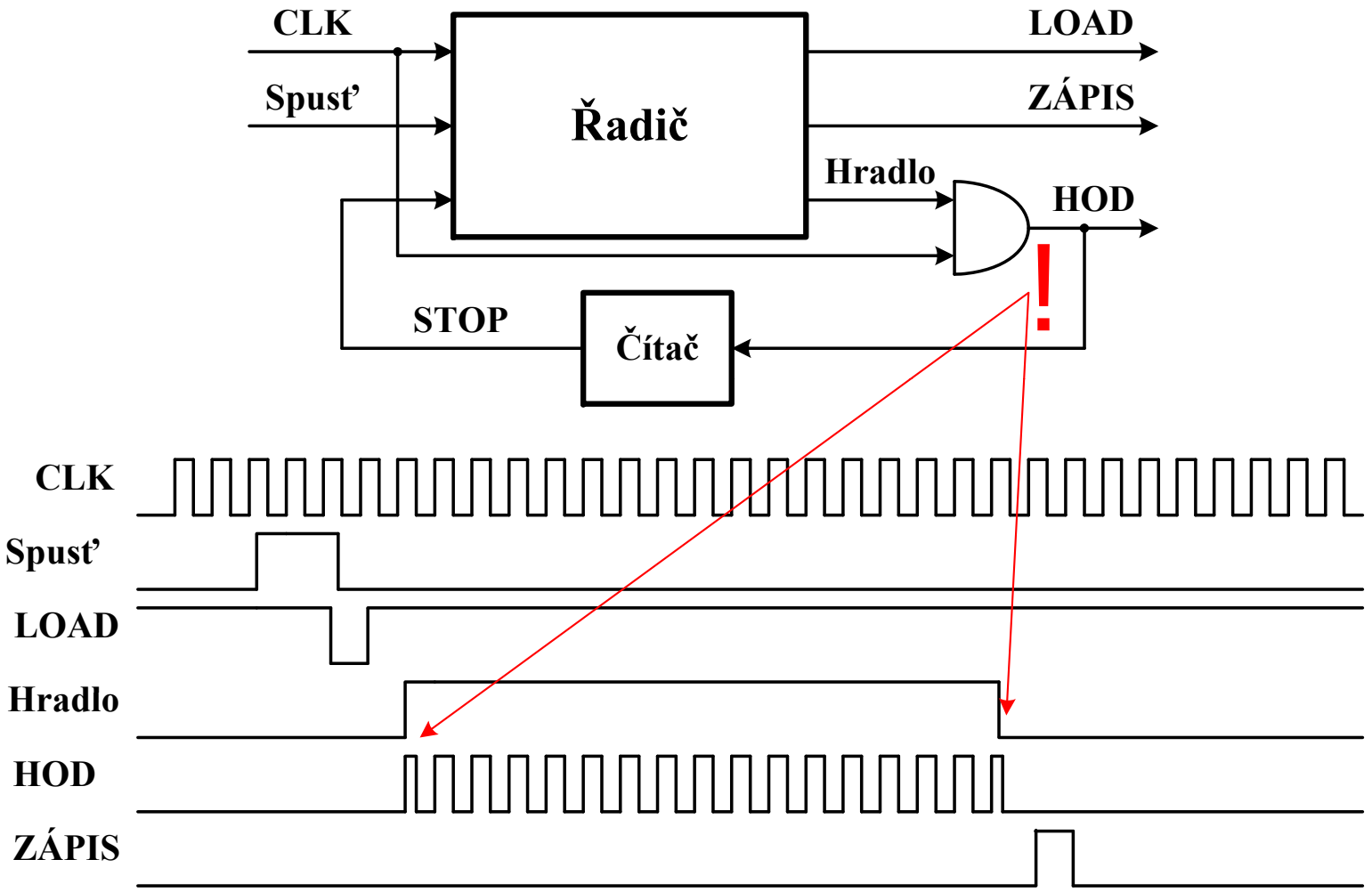
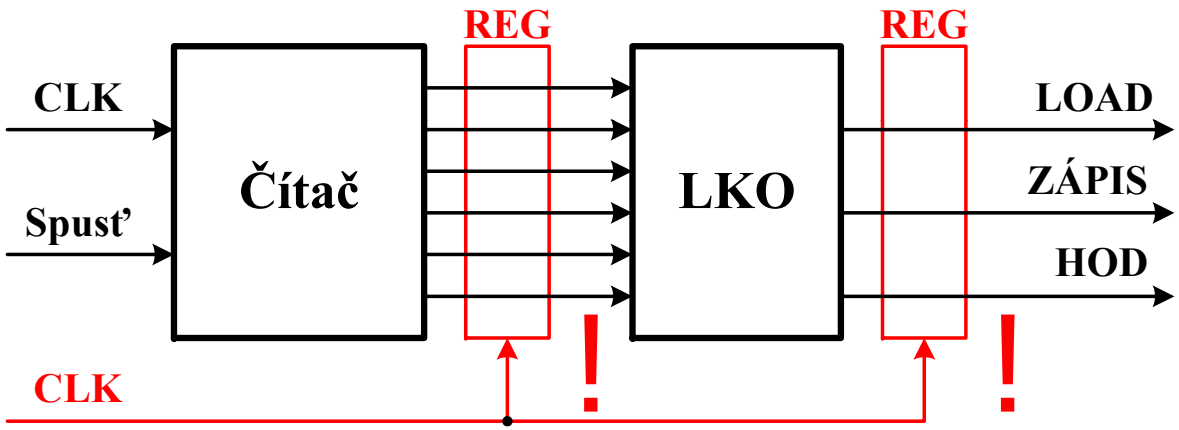
REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.1

Řadič s malým počtem stavů využívající hradlování hodinového signálu. **Výhody**: signál HOD má 2x vyšší kmitočet proti ostatním řešením. **Nevýhody**: Nebezpečí krátkého impulzu HOD.



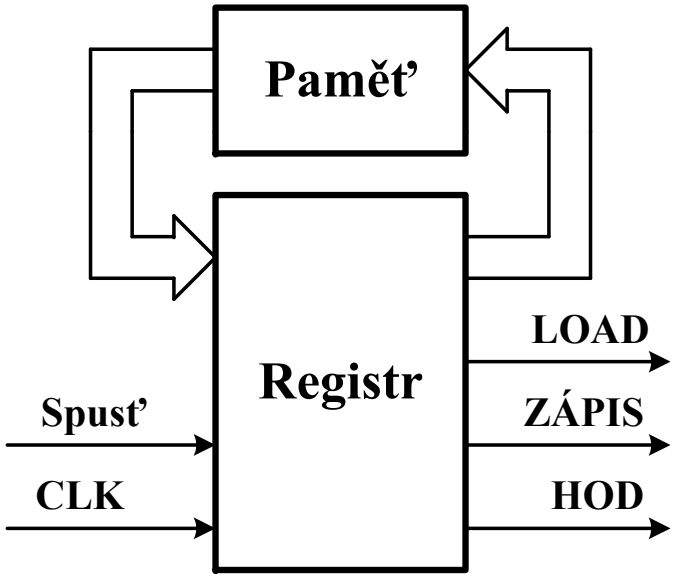
REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.2

Řadič je tvořen čítačem, jehož stavy jsou kombinačním obvodem dekódovány na potřebné řídící signály. **Výhody**: Snadný, byť složitější návrh řídících signálů. **Nevýhody**: Nebezpečí se skrývá ve vlastnostech čítače a postupných změn jeho výstupů (asynchronní/ synchronní). Řídící signály mohou vykazovat statické hazardní stavy. Odstranění případných problémů pomocí vyrovnávacích registrů, řízených stejným hodinovým signálem.



REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.3

Mikroprogramovatelný řadič, realizovaný registrem a pamětí ROM (LKO, RAM). Na každém paměťovém místě paměti je uložen stav řídících signálů (Load, Zápis, Hod) a následující adresy, z které se bude číst nový stav řídících signálů a následující adresy. **Výhody**: Jednoduchý bezproblémový návrh. **Nevýhody**: Vzhledem k velkému počtu hodinových impulzů definovaných dvěma stavy obvodu, je vytvoření obsahu paměti rozsáhlé.



REALIZACE SPOUŠTĚCÍHO IMPULZU SPUSŤ

U všech návrhů se přepokládá spuštění řadiče od změny hodinového signálu zajišťujícího přepínání jednotlivých segmentů. Doba trvání tohoto impulzu v log.1 je velká a je nebezpečí opětovného spuštění řadiče. Z tohoto důvodu bude náběžnou případně sestupnou hranou vytvořen impuls (log.1), který bude ukončen spuštěním řadiče a vytvořením signálu LOAD.

Generování potřebných hodinových signálů

Na přípravku jsou k dispozici dva hodinové signály (10MHz a 50MHz). Z jednoho z nich bude potřeba vytvořit hodinový signál pro přepínání segmentů (Znak_CLK) a hodinový signál pro řadič. V tomto případě předpokládám, že si student z katalogu Texas Instruments si vybere vhodný typ, který se nachází i v knihovně PALASM2.

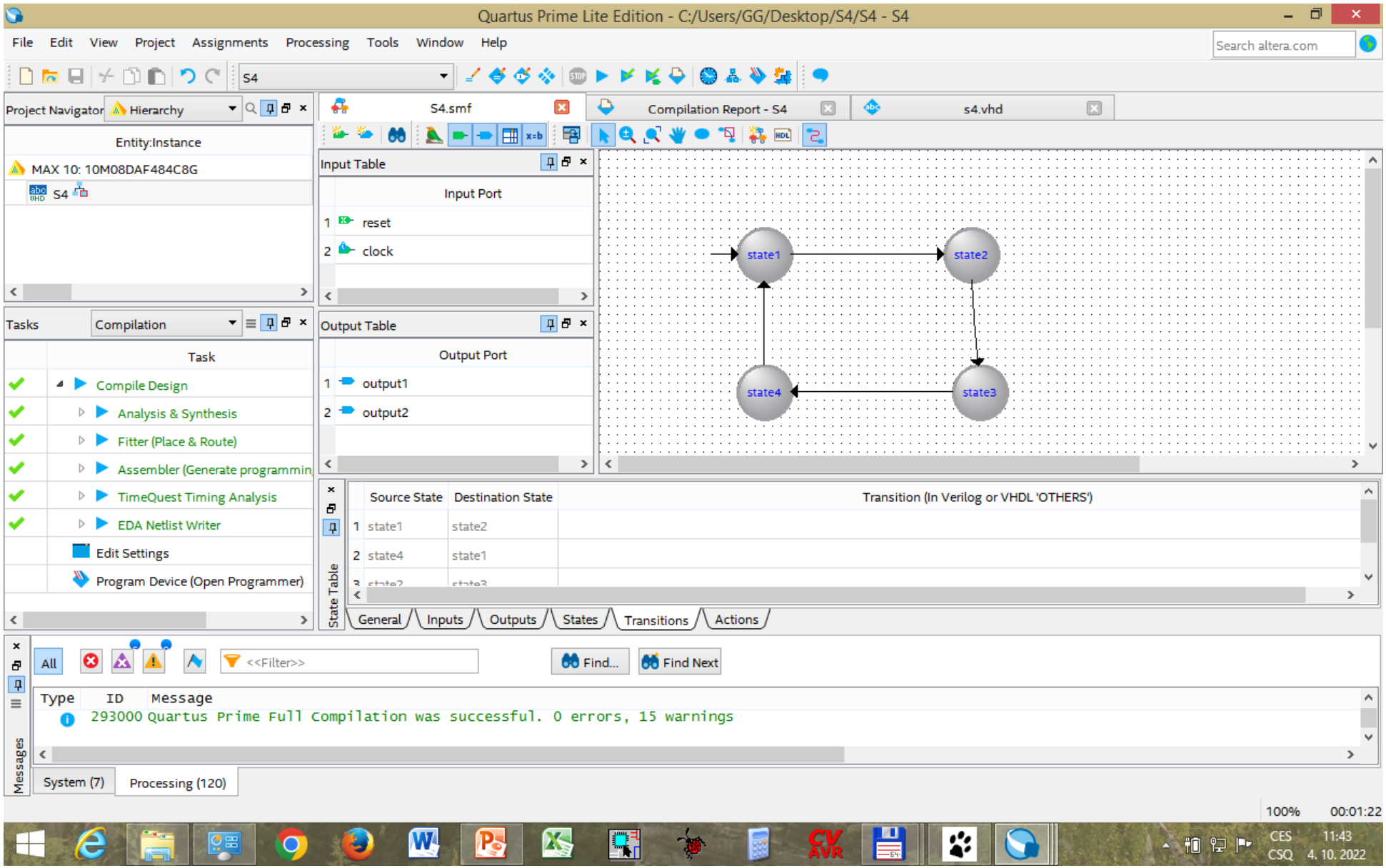
NÁVRH LSO ZE STAVOVÉHO DIAGRAMU

Pro řízení dekodéru i vybavení jednotlivých zobrazovaných hodnot, bude potřeba čítač modulo 4. Ten můžeme navrhnout klasiky s PČ D, JK, T nebo pomocí návrhového systému QuartusII.

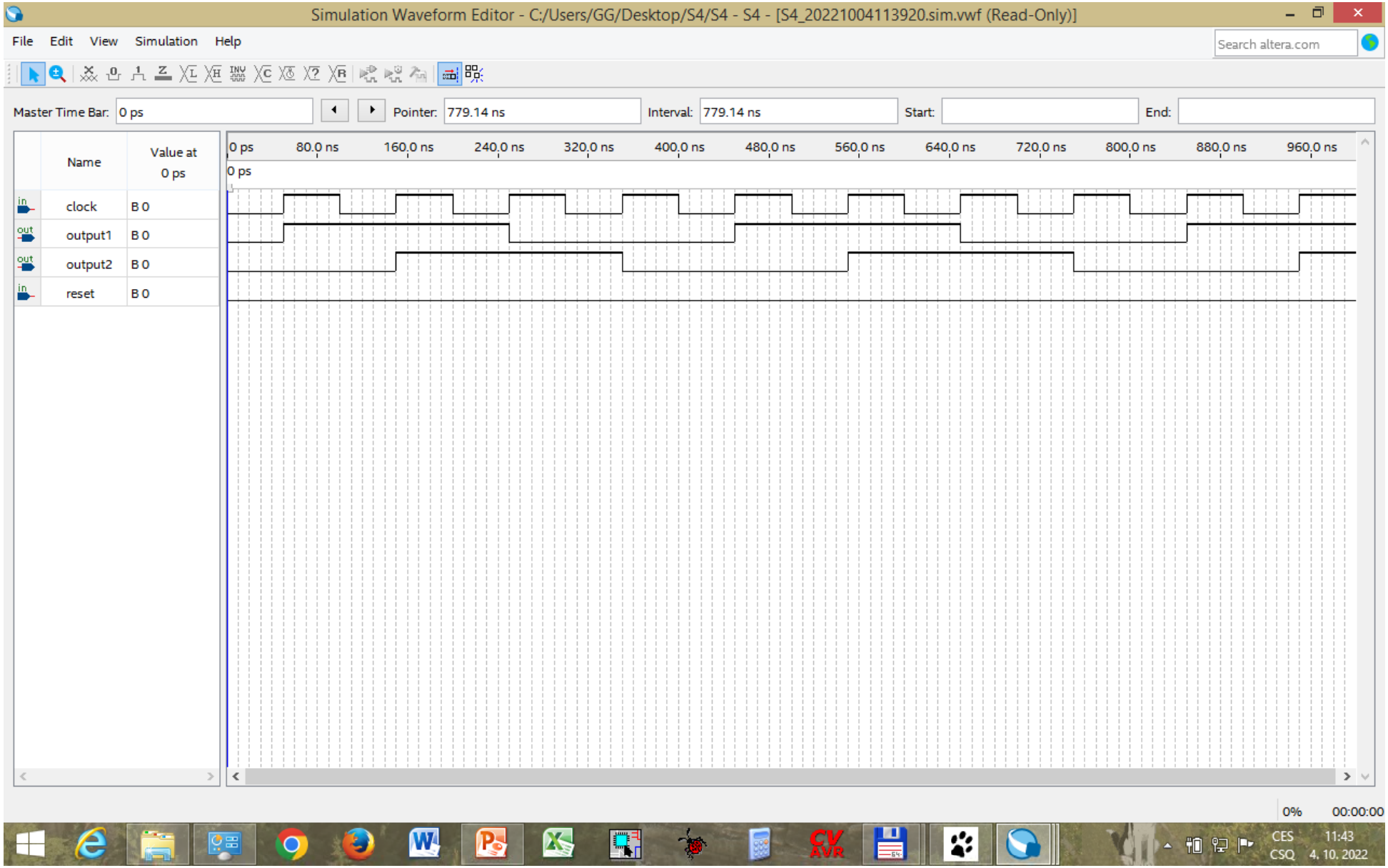
- Založíme nový projekt
- V položce New vybereme State Machine File
- Umístíme na plochu 4 stavy
- Vytvoříme mezi nimi přechody, které v tomto případě budou nepodmíněné.
- Nadefinujeme dvě výstupní proměnné
- Klikneme na jednotlivé stavy a definujeme v položce Action odpovídající stav výstupních proměnných.
- Ikonou HDL necháme vytvořit soubor VHDL
- Necháme zkompilovat vytvořený projekt
- Vytvoříme simulační soubor a ověříme návrh

Více v - Návod na použití programu Quartus.pdf

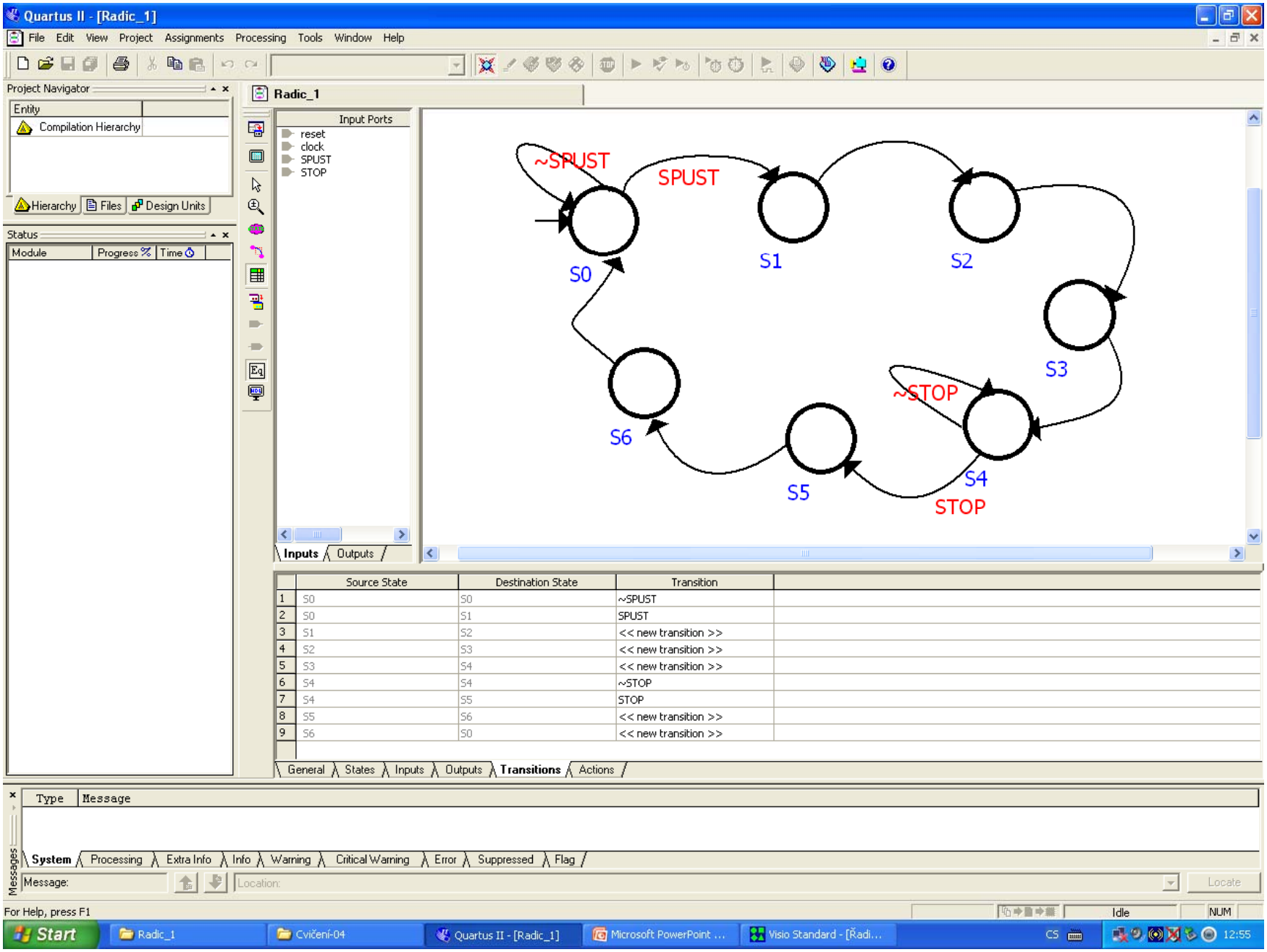
NÁVRH LSO ZE STAVOVÉHO DIAGRAMU



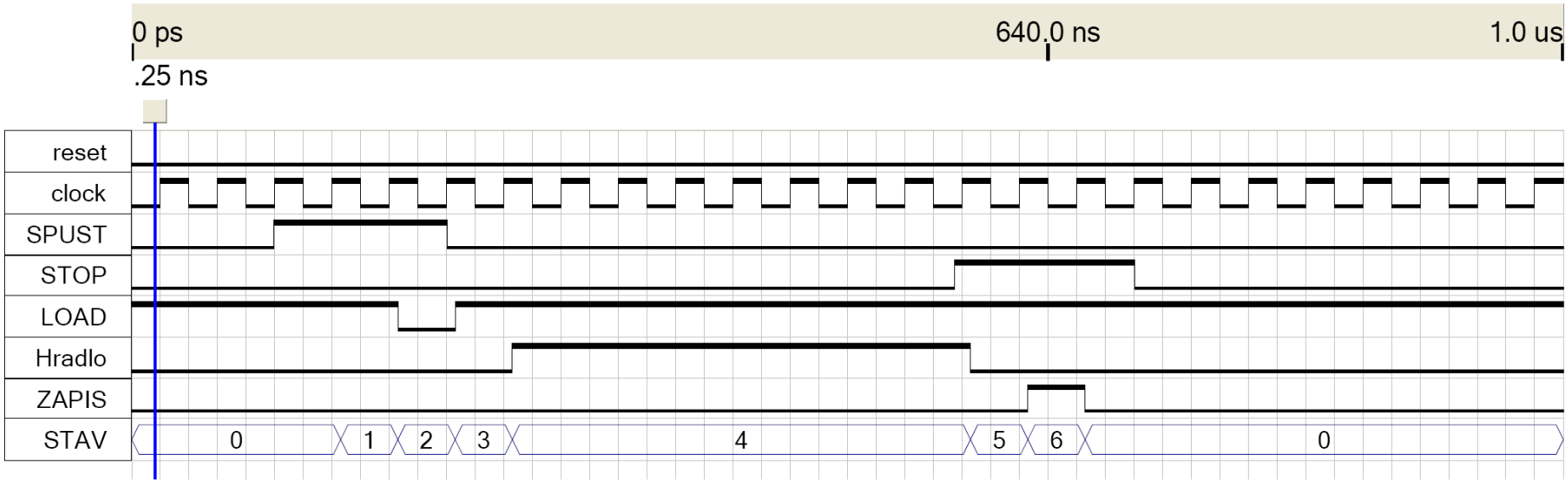
NÁVRH LSO ZE STAVOVÉHO DIAGRAMU



REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.1



REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.1



REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.2

K realizaci bude potřeba:

Prodleva – 1 stav, LOAD – 2 stavy, HOD – 32 stavů, ZÁPIS – 2 stavy

Výstupní funkce vedou na mapy 7 proměnných – trochu problém.

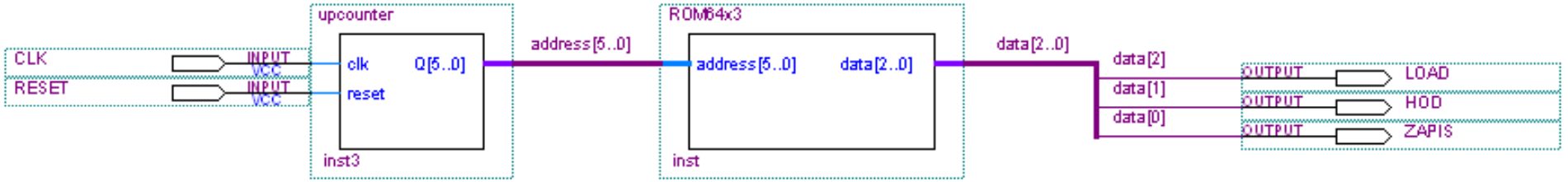
LKO budeme realizovat pamětí ROM.

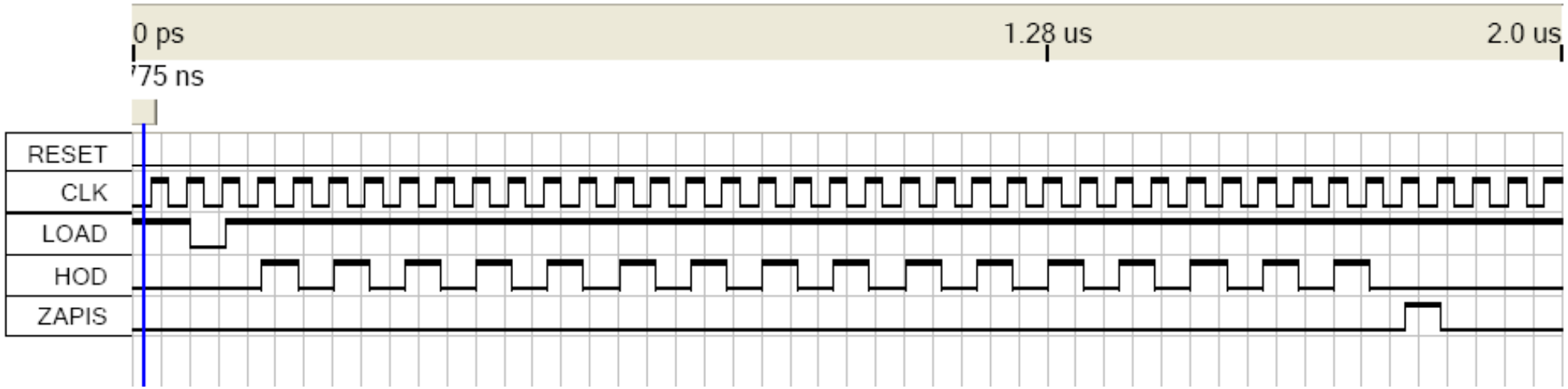
Čítač	LOAD HOD ZÁPIS	Čítač	LOAD HOD ZÁPIS	Čítač	LOAD HOD ZÁPIS	Čítač	LOAD HOD ZÁPIS	Čítač	LOAD HOD ZÁPIS
0	1 0 0	8	1 1 0	16	1 1 0	24	1 1 0	32	1 1 0
1	1 0 0	9	1 0 0	17	1 0 0	25	1 0 0	33	1 0 0
2	0 0 0	10	1 1 0	18	1 1 0	26	1 1 0	34	1 1 0
3	1 0 0	11	1 0 0	19	1 0 0	27	1 0 0	35	1 0 0
4	1 1 0	12	1 1 0	20	1 1 0	28	1 1 0	36	1 0 1
5	1 0 0	13	1 0 0	21	1 0 0	29	1 0 0	37	1 0 0
6	1 1 0	14	1 1 0	22	1 1 0	30	1 10	38	1 0 0
7	1 0 0	15	1 0 0	23	1 0 0	31	1 0 0	39	1 0 0

REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.2

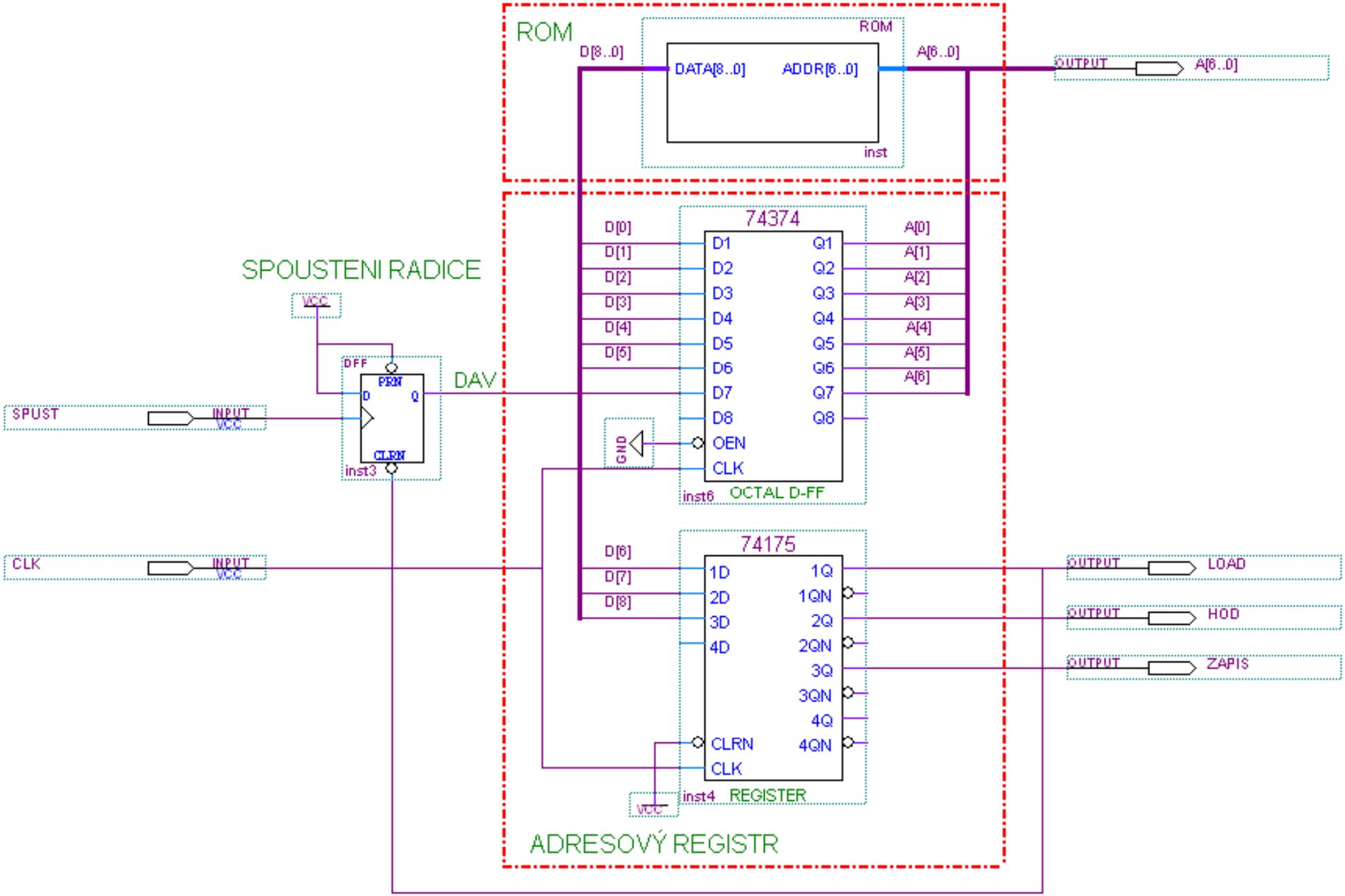
```
library IEEE;
use IEEE.std_logic_1164.all;

entity ROM64x3 is
port (address : in INTEGER range 0 to 63;
     data : out std_logic_vector (2 downto 0));
end entity ROM64x3;

architecture LKO of ROM64x3 is
type rom_array is array (0 to 63) of std_logic_vector (2 downto 0);
constant rom : rom_array :=
   ("100","100","000","100","110","100","110","100","110","100","110","100",
    "110","100","110","100","110","100","110","100","110","100","110","100",
    "110","100","110","100","110","100","110","100", "110","100","110","100",
    "101","100","100","100„,"100","100","100","100","100","100","100","100",
    "100","100","100","100","100","100","100","100",
    "100","100","100","100","100","100","100","100");
    begin
       data <= rom(address);
end architecture LKO;
```

REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.2





REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.3



REALIZACE ŘADIČE PRO DYNAMICKY OVLÁDANÝ DISPLEJ - VAR.3