Decodificator BCD la 7 Segmente (Verilog)

Acest proiect implementează un decodificator combinat care transformă un semnal de intrare de 4 biți în format BCD (Binary Coded Decimal) într-un semnal de ieșire de 7 biți pentru controlul unui afișaj cu 7 segmente (catod comun).

Structura Proiectului

- `bcd_to_7seg.v` - Modulul principal care conține logica de decodificare.
- `tb_bcd_to_7seg.v` - Testbench pentru verificarea funcționalității și afișarea cifrelor în terminal.
- `wave.vcd` - Fișierul de forme de undă generat în urma simulării.

Implementare

bcd_to_7seg.v

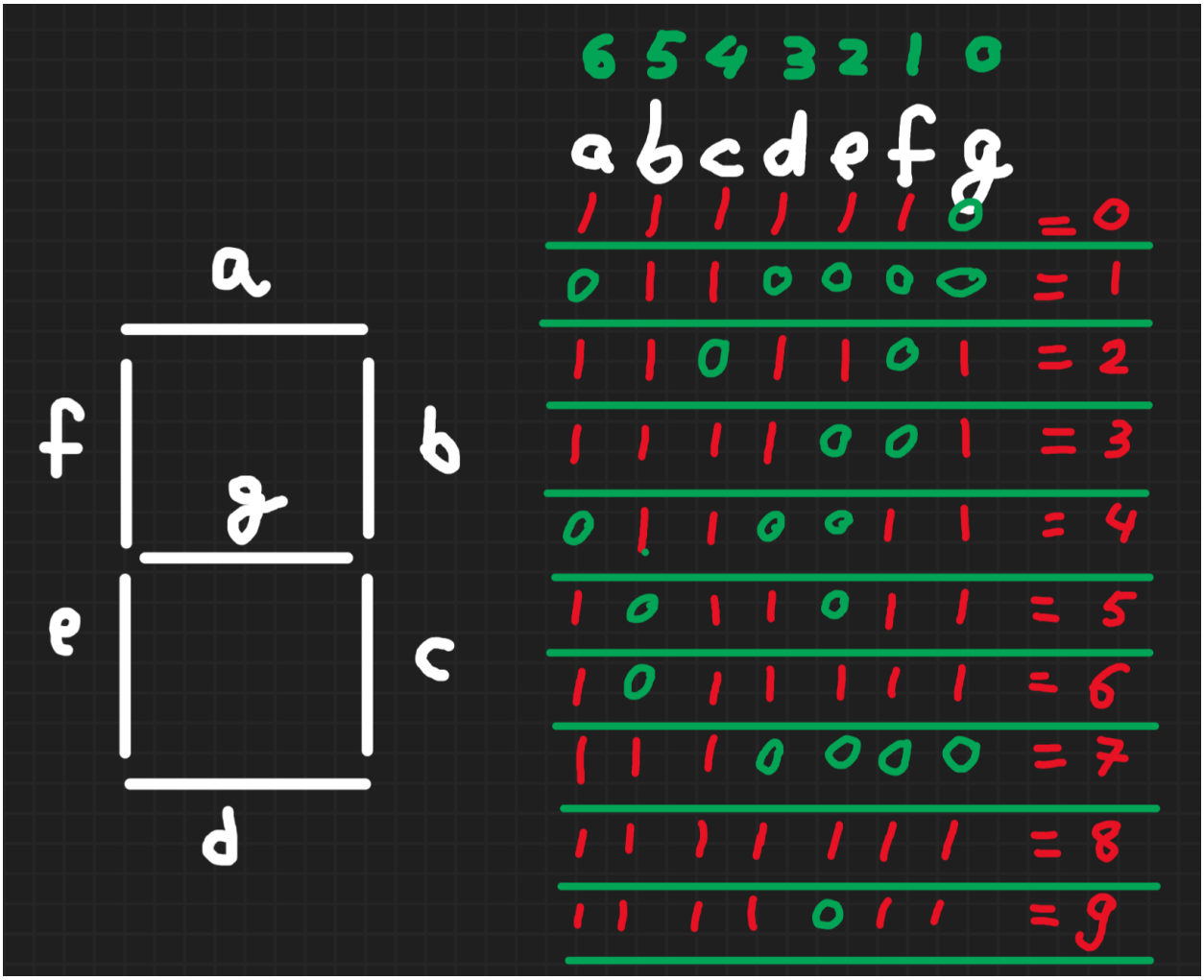
Modulul primește o intrare de 4 biți (`bcd`) și produce:

- `seg[6:0]` - Starea celor 7 segmente (a, b, c, d, e, f, g).
- `valid` - Un semnal de control care indică dacă intrarea este o cifră BCD validă (0-9).

Tabel de Adevăr (Logic):

BCD (In)	Cifră	Segmente (abcdefg)	Valid
0000	0	1111110	1
0001	1	0110000	1
0010	2	1101101	1
0011	3	1111001	1
0100	4	0110011	1

BCD (In)	Cifră	Segmente (abcdefg)	Valid
0101	5	101101	1
0110	6	1011111	1
0111	7	1110000	1
1000	8	1111111	1
1001	9	1111011	1
>1001	E	1001111	0



Logica de Control

Implementarea folosește un bloc `always @(*)` cu o instrucțiune `case` pentru a asigura un comportament pur combinațional. Pentru valorile de intrare cuprinse între 10 și 15 (care nu sunt valide în codificarea BCD), modulul setează semnalul `valid` pe `0` și afișează un simbol de eroare.

Instrucțiuni de Rulare

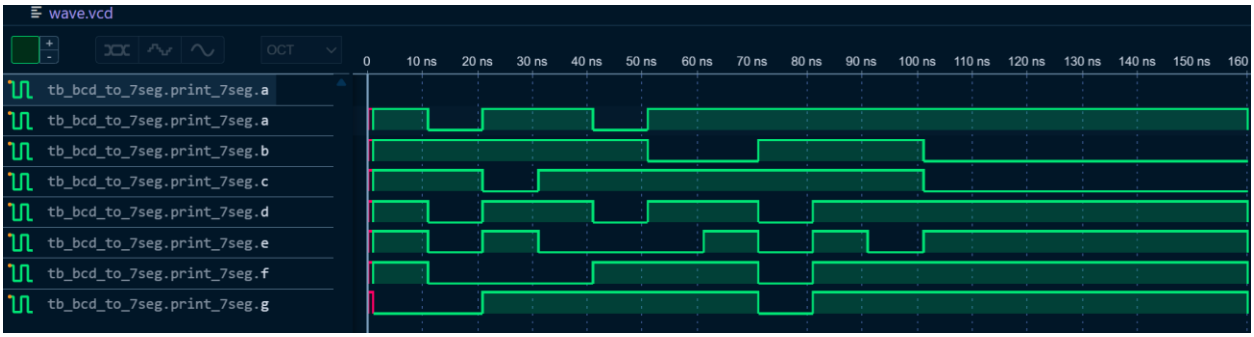
Pentru a rula simularea, este necesar un simulator Verilog (ex: Icarus Verilog).

1. **Compilare:**
2. `iverilog -o sim bcd_to_7seg.v tb_bcd_to_7seg.v`
3. **Execuție:**
4. `vvp sim`
5. **Vizualizare Forme de Undă:** Fișierul `wave.vcd` poate fi deschis cu GTKWave sau o extensie VCD în VS Code.

Analiza Diagramelor de Semnal

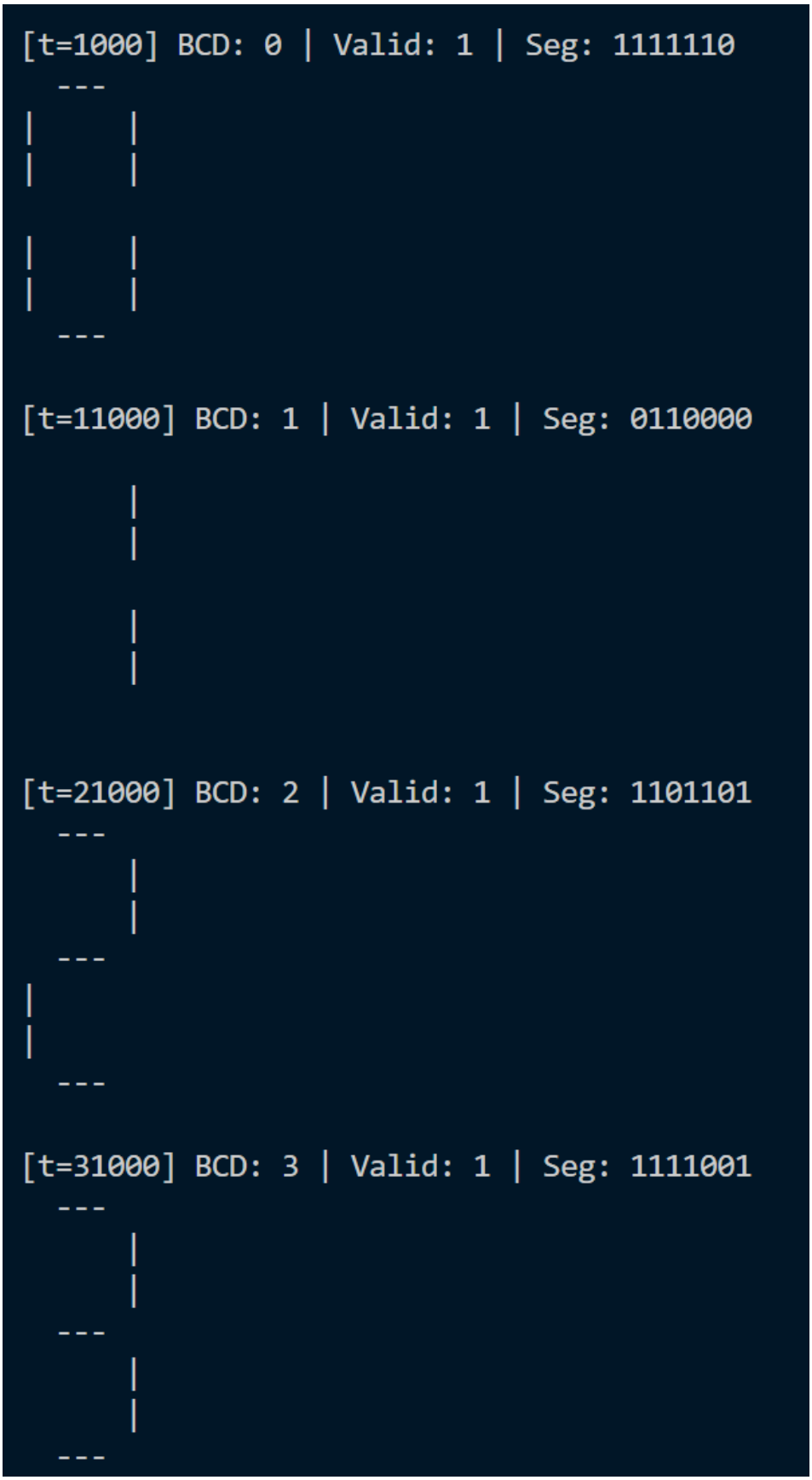
În urma simulării, se pot observa următoarele comportamente în diagrama de timp:

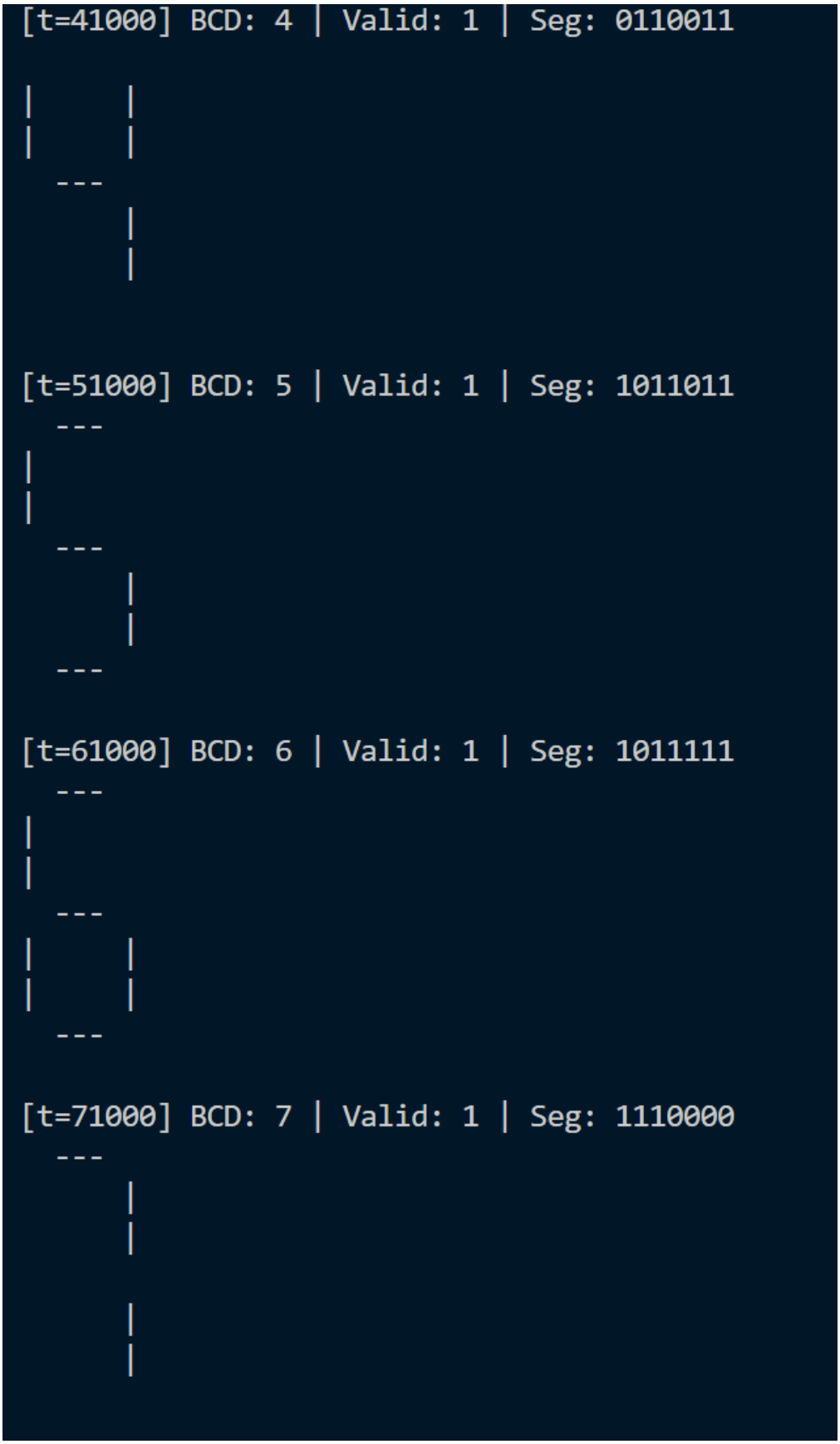
- **Axa timpului**: Fiecare cifră este testată la un interval de 10ns.
- **Semnalul `valid`**: Acesta stă în starea logică `1` pentru intrările `0-9` și trece în `0` imediat ce intrarea depășește valoarea `9`.
- **Semnalul `seg`**: Codurile hexazecimale corespund segmentelor aprinse pentru fiecare cifră (ex: `7E` pentru cifra 0).



Rezultate

După execuția programului, putem observa rezultatele obținute, respectiv tratarea cazurilor de excepție:


[t=1000] BCD: 0 | Valid: 1 | Seg: 1111110
[t=11000] BCD: 1 | Valid: 1 | Seg: 0110000
[t=21000] BCD: 2 | Valid: 1 | Seg: 1101101
[t=31000] BCD: 3 | Valid: 1 | Seg: 1111001


[t=41000] BCD: 4 | Valid: 1 | Seg: 0110011
[t=51000] BCD: 5 | Valid: 1 | Seg: 1011011
[t=61000] BCD: 6 | Valid: 1 | Seg: 1011111
[t=71000] BCD: 7 | Valid: 1 | Seg: 1110000

```
[t=81000] BCD: 8 | Valid: 1 | Seg: 1111111
  ---
|     |
|     |
  ---
|     |
|     |
  ---

[t=91000] BCD: 9 | Valid: 1 | Seg: 1111011
  ---
|     |
|     |
  ---
      |
      |
  ---

[t=101000] BCD: 10 | Valid: 0 | Seg: 1001111
  ---
|
|
  ---
|
|
  ---

[t=111000] BCD: 11 | Valid: 0 | Seg: 1001111
  ---
|
|
  ---
|
|
  ---
```

```
[t=121000] BCD: 12 | Valid: 0 | Seg: 1001111
  ---
|
|
  ---
|
|
  ---

[t=131000] BCD: 13 | Valid: 0 | Seg: 1001111
  ---
|
|
  ---
|
|
  ---

[t=141000] BCD: 14 | Valid: 0 | Seg: 1001111
  ---
|
|
  ---
|
|
  ---

[t=151000] BCD: 15 | Valid: 0 | Seg: 1001111
  ---
|
|
  ---
|
|
  ---
```