**Step 1: Define Inputs and Outputs**

We are designing a **3-bit Up/Down Counter** using **D Flip-Flops**.  
Inputs:

* **Q2, Q1, Q0** → Present state (current output of flip-flops)
* **Up/Down (U)** → Mode selection (0 for UP, 1 for DOWN)

Outputs:

* **D2, D1, D0** → Next state (input to flip-flops)

**Step 2: Create the State Table**

| **U (Up/Down)** | **Q2 Q1 Q0** | **Next State (UP)** | **Next State (DOWN)** |
| --- | --- | --- | --- |
| 0 | 000 | 001 | 111 |
| 0 | 001 | 010 | 000 |
| 0 | 010 | 011 | 001 |
| 0 | 011 | 100 | 010 |
| 0 | 100 | 101 | 011 |
| 0 | 101 | 110 | 100 |
| 0 | 110 | 111 | 101 |
| 0 | 111 | 000 | 110 |
| 1 | 000 | 111 | 001 |
| 1 | 001 | 000 | 010 |
| 1 | 010 | 001 | 011 |
| 1 | 011 | 010 | 100 |
| 1 | 100 | 011 | 101 |
| 1 | 101 | 100 | 110 |
| 1 | 110 | 101 | 111 |
| 1 | 111 | 110 | 000 |

**Step 3: Extract Flip-Flop Input Values (D2, D1, D0)**

For **D Flip-Flops**,

D=QnextD = Q\_{\text{next}}D=Qnext​

From the state table:

| **U** | **Q2 Q1 Q0** | **D2 (UP)** | **D1 (UP)** | **D0 (UP)** | **D2 (DOWN)** | **D1 (DOWN)** | **D0 (DOWN)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 000 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 001 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 010 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 011 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 100 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 101 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 110 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 111 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 000 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 001 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 010 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 011 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 100 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 101 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 110 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 111 | 1 | 1 | 0 | 0 | 0 | 0 |

**Step 4: Karnaugh Maps (K-Maps) and Reduction**

**D0 K-map**

| **Q1Q0** | **00** | **01** | **10** | **11** |
| --- | --- | --- | --- | --- |
| **U=0, Q2=0** | 0 | 1 | 0 | 1 |
| **U=0, Q2=1** | 0 | 1 | 0 | 1 |
| **U=1, Q2=0** | 1 | 0 | 1 | 0 |
| **U=1, Q2=1** | 1 | 0 | 1 | 0 |

Reduced Equation:

D0=Q0⊕1D0 = Q0 \oplus 1D0=Q0⊕1

This is the same as:

vhdl

CopyEdit

D0 <= NOT Q0;

**D1 K-map**

| **Q1Q0** | **00** | **01** | **10** | **11** |
| --- | --- | --- | --- | --- |
| **U=0, Q2=0** | 0 | 0 | 1 | 1 |
| **U=0, Q2=1** | 0 | 0 | 1 | 1 |
| **U=1, Q2=0** | 1 | 1 | 0 | 0 |
| **U=1, Q2=1** | 1 | 1 | 0 | 0 |

Reduced Equation:

D1=Q1⊕(Q0⊕U)D1 = Q1 \oplus (Q0 \oplus U)D1=Q1⊕(Q0⊕U)

VHDL Form:

vhdl

CopyEdit

D1 <= (Q1 XOR Q0) WHEN U = '0' ELSE (Q1 XOR NOT Q0);

**D2 K-map**

| **Q1Q0** | **00** | **01** | **10** | **11** |
| --- | --- | --- | --- | --- |
| **U=0, Q2=0** | 0 | 0 | 0 | 1 |
| **U=0, Q2=1** | 1 | 1 | 1 | 0 |
| **U=1, Q2=0** | 1 | 1 | 1 | 0 |
| **U=1, Q2=1** | 0 | 0 | 0 | 1 |

Reduced Equation:

D2=Q2⊕((Q1⊕U)⋅Q0)D2 = Q2 \oplus ((Q1 \oplus U) \cdot Q0)D2=Q2⊕((Q1⊕U)⋅Q0)

VHDL Form:

vhdl

CopyEdit

D2 <= (Q2 XOR ((Q1 XOR U) AND Q0)) WHEN U = '0' ELSE (Q2 XOR ((NOT (Q1 XOR U)) AND Q0));

**Final VHDL Equations**

vhdl

CopyEdit

D0 <= NOT Q0;

D1 <= (Q1 XOR Q0) WHEN U = '0' ELSE (Q1 XOR NOT Q0);

D2 <= (Q2 XOR ((Q1 XOR U) AND Q0)) WHEN U = '0' ELSE (Q2 XOR ((NOT (Q1 XOR U)) AND Q0));

These equations are optimized using **K-maps** and will correctly implement a **3-bit Up/Down Counter** in **VHDL**.