**Hierarchy**

**Comparator**

|  |  |
| --- | --- |
| **Input** | **Output** |
| clk | isMatch |
| rst |  |
| stop |  |
| dbValid |  |
| [21:0] inQuery |  |
| [21:0] inDB |  |
| highBitEnd |  |

Compares two inputs of query wmer ***[21:0] inQuery*** and database wmer ***[21:0] inDB****.* The output ***isMatch***is high if two wmers exactly match. ***isMatch***gets low when ***rst*** is high or ***stop*** and ***highBitEnd*** signals come.

**Shift2Reg**

|  |  |
| --- | --- |
| **Input** | **Output** |
| clk | [511:0] outData |
| rst |
| load |
| shift |
| [511:0] inData |
| dataValid |

Shifts input data ***[511:0] inData*** depending on the input signals ***load*** and ***shift.*** Then shifted input data is provided by output data ***[511:0] outData***.

**Hit**

|  |  |
| --- | --- |
| **Input** | **Output** |
| clk | [8:0] ShiftNo |
| rst | wire hit |
| [511:0] query | reg startExpand |
| queryValid | [8:0] locationQ |
| [511:0] dataBase | reg highBitEnd |
| dataBaseValid |  |
| shift |  |
| load |  |
| stop |  |

Waits for ***stop***

Waits for ***hit*** and ***dataBaseValid***, then If (ouput[k]==1) sends signal ***startExpand*** and goes to the next state

The ***Hit.v*** module consists of two states. The state Idle is needed for waiting for the ***hit*** and ***dataBaseValid*** signals, which comes if at least one bit is high in ***ouput*** register. When hit and ouput[k] registers get high (where ***k*** is a register that shows the position of current high bit that is being considered in the ***ouput*** register) it makes ***startExpand*** signal high and goes to ***HitLow*** state which waits for the stop signal that comes from the ***ExpandFSM.v*** module.

//After, it increments the k register then goes to ***Idle*** state again, to search next exact match.

**ExpandFSM.v**

|  |  |
| --- | --- |
| **Input** | **Output** |
| clk | reg load, |
| rst | [31:0] outAddress, |
| start | reg [31:0] locationStart, |
| queryValid | reg [31:0] locationEnd, |
| dataValid | [10:0] Score, |
| [8:0] shiftNo | reg stop |
| [16:0] dataCounter |  |
| [511:0] inQuery |  |
| [8:0] LocationQ |  |
| [511:0]inDB |  |
| loadDone |  |

\

if(dataMerged[m1-:2] != Query[i1-:2] & dataMerged[m2+:2] != Query[i2+:2])

else if(k1 == range1 & k2 == range2)

0

waits for dataVaid signal,

loads 2nd data

0

If(!stop & start)

load <= 1'b1;

0

waits for loadDone signal,

load <= 1'b0;

0

waits for loadDone signal,

load <= 1'b0;

0

If(shiftNumber > 290)

0

If(shiftNumber < 199)

0

If(dataCounter == 0 & shiftNumber < 199)

***highScore.v***

|  |  |
| --- | --- |
| **Input** | **Output** |
| clk | [10:0] Score |
| rst, |  |
| [1:0] b1 |  |
| [1:0] b2 |  |
| startCalc |  |

***highScore.v*** is a module instantiated in ExpandFsm.v. The module is utilized for calculating score while expansion is being processed.

***memInt.v***

|  |  |
| --- | --- |
| **Input** | **Output** |
| clk | reg ddr\_rd |
| rst | reg [`ddrAddrWidth-1:0] readAdd |
| ddr\_rd\_done | locationStart |
| ddr\_rd\_valid | locationEnd |
| [511:0] ddr\_rd\_data | hitTEST |
| [511:0] query |  |
| queryValid |  |