CHAPTER 6

SEMI SERIAL ON-CHIP ROUTER

6.1 INTRODUCTION

In this work, introduce a low power and high performance serial on-chip communication link based on innovative design techniques and its design methodologies are presented in this research work.

Ethiopia Nigussie et al (2012) have proposed a high-throughput and low-energy semi-serial on-chip communication link based on novel design techniques and circuit solutions is presented. This self-timed link is designed using high-speed serialization/deserialization and pulse dual-rail encoding techniques. The link also employs wave-pipelined differential pulse current-mode signaling to maintain the high speed data intake from the serializer. The energy efficiency of their proposed semi-serial link, which consists of bit-serial links in parallel, mainly comes from the sharing of the novel serializer’s control circuit among the bit-serial links. In addition, the integration of pulse signaling with wave-pipelining, the use of a new low-complexity data validity detection technique, and the avoidance of data decoding logic also contribute to the power reduction.

The proposed semi-serial link is designed using high speed serialization/deserialization and multi-orthogonal encoding techniques. The link also employs acknowledgement scheme to maintain the high speed data intake from the serializer. The proposed semi-serial link is analyzed and
compared with bit-serial and fully bit-parallel links for 64-bit data communications. The semi serial link has a single serial line which has the transmission capability of eight bits per cycle for 8mm transmission range. But, the parallel link has a width of eight bit on its own transmission line which requires single clock pulse for the transmission of eight bits. Hence, the semi serial on chip link consumes lowest energy consumption. The proposed semi-serial on-chip is designed and simulated in Xilinx Project navigator and tested on various FPGA devices using CMOS technology. Semi-serial on-chip communication link based on novel design techniques and circuit solutions is presented. This self-timed link is designed using high-speed serialization/deserialization and pulse dual-rail encoding techniques. The link also employs wave-pipelined differential pulse current-mode signaling to maintain the high speed data intake from the serializer.

The power consumptions of the 32-bit serializer and deserializer were 2.198 and 1.416 mW, respectively. The power consumption of the link and its energy dissipation per bit did not increase steeply with the wire length because of the pulse signaling, where only a small portion of the wire needs to be charged. The power consumption of the clock buffer and tree were 0.576 and 0.035 mW, respectively.

The proposed methodology is a low power and high performance serial on-chip link which includes sending and receiving routers, serializer, deserializer, multi-orthogonal encoder and decoder. Data is sent from source router to input channel of the communication node through serializer and encoder, and then it is transmitted to the corresponding output router. The proposed serial on-chip link deals with the problems of collision and limited power transmission.
6.2 RELATED WORKS ON SEMI SERIAL ON-CHIP ROUTER

In Faruque et al (2008) an adaptive architecture with runtime observability has been proposed to avoid faults in NoCs, providing adaptability both at the system-level and at the architecture-level. At system-level the architecture can re-map the system tasks and at architecture-level it can re-route the packets and re-allocate the virtual channel buffers (VCB). The presence of faults triggers the need of NoC adaptation at architecture-level. The changes at architecture-level are based on the occurrence of faults, and these events occur when packets do not reach the destination or when the VCB is full. This adaptive process occurs only when a fault occurs, and hence no performance or power advantage can be obtained during the normal operation of the system.

In Jose (2005), a serial on-chip link has been designed by adopting circuits that had originally been used for off-chip communications. It uses output multiplexed transmitter architecture due to its ability to deliver better performance than input multiplexing. However, this comes at the expense of a much higher output capacitance that grows linearly with the bit-width. Both transmitter and receiver use multi-phase DLL circuits and clock calibration is required at the receiver side. They have fabricated a prototype chip in 180nm CMOS technology and a 3mm link has achieved a throughput of 8Gbps.

A high-speed serial link was presented in Lee (2005). The serializer/deserializer is based on chain of Multiplexers. The link is single-ended and employs wave-pipelining. As a timing reference, constant delay elements are used instead of a clock. Furthermore, the operation is based on the assumption that the introduced unit delays for the serializer and deserializer are the same. However, getting the same delay is almost impossible in the sub-100nm CMOS technology due to considerable Process,
voltage and temperature (PVT) variations, which in turn affect the reliability of communication. A test chip was manufactured using 180nm CMOS technology and the measured throughput was 3Gbps.

In Duato (1997), various Interconnection Networks were designed and evaluated in different network routing protocols. The results were analyzed with existing networks.

In this work Matos (2010), they have designed a reconfigurable routers for wireless media access. But, this work was entirely based on four ports only and complex architecture in its basic internal structure. It also consumed more than 78% hardware utilization.

In this work Park,D et al (2006), they have designed a Virtual Channel Regulator (VCR) for Network-on-Chip routers. This paper dealt different configuration protocols in a linear manner and results were compared against Virtual buffer regulator of routers.

In Liu (2007), MANET’s distributed architecture and changing topology, and a traditional centralized monitoring technique were analyzed in MANETs acknowledgment based approach for the detection of routing misbehaviour.

6.3 PROPOSED LINK ARCHITECTURE

A basic communication link of serial on-chip link includes sending and receiving routers, serializer, deserializer, multi-orthogonal encoder and decoder. Data is sent from source router to input channel of the communication node through serializer and encoder, and then it is transmitted to the corresponding output router. The block diagram of our proposed serial
on-chip link router is shown in Figure 6.1 and is explained in detail in the next section.

![Figure 6.1 Overall Block Diagram of Proposed Serial on-Chip Link Design](image_url)

**6.3.1 Sending Side Router**

The router at the sending side includes buffers, switches and control units required for storing and forwarding data from the input ports to the preferred output port. The router comprises a smaller area and buffer size and is shown in Figure 6.2. The router architecture has several internal modules which include Packet Counter Module (PCM), Address Counter Module (ACM), Address Differentiation Module (ADM) and Virtual Channel Buffer (VCB).
Figure 6.2 Router Architecture at Sending Side

The WXY routing technique is followed as our routing methodology. Consider the tuple \( \Gamma = \{N, S, W, E, NE, SE, SW, NW, D_{IN}\} \). Each direction \( d \in \Gamma \) has a weight \( W_d \) and available bandwidth \( B_d \) with \( B_d \leq B_{max} \), and \( B_{max} \) is the maximum link capacity. The current router coordinates are \( (x, y) \) and every packet \( D_{IN} \) has destination coordinates \( x_d, y_d \) and a required bandwidth \( B_p \). The weights are assigned using the equations.

6.3.2 Multi-orthogonal Encoder and Decoder

The block diagram of the encoder is shown in Figure 6.6 and consists of multi code generator, orthogonal multiplexers- \( i, j, k, l \) and orthogonal multipliers. The constant amplitude encoder possesses 3 parity bits namely, \( L^*, L_1, L_0 \) generated from 3 groups of parallel bits \( (i^*, i_1, i_0), (j^*, j_1, j_0), (k^*, k_1, k_0) \) according to the following equation:
\[ L^* = I^* \land j^* \land k^* \]
\[ L_i = i_i \land j_i \land k_i \]
\[ L_0 = i_0 \land j_0 \land k_0 \]

The output of the orthogonal multiplexers \( i, j, k, l \) are given by,

\[ i = (0,0,i_1,i_0) = (0,0,1,0) \]
\[ j = (0,1,j_1,j_0) = (0,1,0,0) \]
\[ k = (1,0,k_1,k_0) = (1,0,0,0) \]
\[ l = (1,1,l_1,l_0) = (1,1,1,0) \]

---

**Figure 6.3 Multi-Orthogonal Encoder Block Diagram**

The multi code generator consists of two blocks, namely, serial to parallel converter and gold sequence generator. The serial to parallel converter converts the data bits into the number of branches according to the length of the gold sequence, the gold sequence can be generated by applying
XOR operation over a few Pseudo random Noise (PN) sequences as explained in Figure 6.4.

![Figure 6.4 Generation of Gold Sequence](image)

### 6.3.3 PN (Pseudo Random Noise) Sequence Generation Circuit

PN sequences are the periodic binary sequences that can be generated using linear feedback shift registers. A feedback shift register consists of an shift registers made up of ‘m’ flip flops and a logic circuit that are interconnected to form the multi-loop feedback circuit.

![Figure 6.5 Linear Feedback Shift Register (LFSR) Generator](image)
The PN Sequence generator circuit initially consists of three ‘D’ flip flops and each can be connected as shown in the figure. For the length of 7 bits, 3 flip flops are used for the generation of PN Sequences.

- for length-$m$ registers they produce a sequence of length $2^{m-1}$.
- The duty cycle of the PN Sequence is $c=[M-1]/[N-1]$;

where, $N=$Length of PN Sequence,
$M=$Number of ones in PN-Sequence.

- Balance Property:

- Of the $(2^m)-1$ terms, $2^{(m-1)}$ are one and $2^{(m-1)}-1$ are zero.

**Table 6.1 Test bncPattern Generation**

<table>
<thead>
<tr>
<th>CLK</th>
<th>QA^QC</th>
<th>Q-A</th>
<th>Q-B</th>
<th>Q-C</th>
<th>STWK-SEQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
6.3.4 Gold Sequence

Gold sequences have been proposed by Gold in 1967 and 1968. These are constructed by EXOR-ing two $m$-sequences of the same length with each other. Thus, for a Gold sequence of length $m = 2^l - 1$, one uses two LFSR, each of length $2^l - 1$. If the LSFRs are chosen appropriately, Gold sequences have better cross-correlation properties than maximum length LSFR sequences.

A Gold code, also known as Gold sequence, is a type of binary sequence, used in telecommunication and satellite navigation. Gold codes are named after Robert Gold. Gold codes have bounded small cross-correlations within a set, which is useful when multiple devices are broadcasting in the same frequency range. A set of Gold code sequences consists of $2^n - 1$ sequences each one with a period of $2^n - 1$.

A set of Gold codes can be generated with the following steps. Pick two maximum length sequences of the same length $2^n - 1$ such that their absolute cross-correlation is less than or equal to $2^{(n+2)/2}$, where $n$ is the size of the Linear Feedback Shift Register (LFSR) used to generate the maximum length sequence (Gold '67). The set of the $2^n - 1$ exclusive-or of the two sequences in their various phases (i.e. translated into all relative positions) is a set of Gold codes. The highest absolute cross-correlation in this set of codes is $2^{(n+2)/2} + 1$ for even $n$ and $2^{(n+1)/2} + 1$ for odd $n$.

The exclusive or of two Gold codes from the same set is another Gold code in some phase

Within a set of Gold codes about half of the codes are balanced – the number of ones and zeros differs by only one.
6.3.5 Preferred Sequences

Gold and Kasami showed that for certain well-chosen $m$-sequences, the cross correlation only takes on three possible values, namely $-1$, $-t$ or $t-2$. Two such sequences are called preferred sequences. Here $t$ depends solely on the length of the LFSR used. In fact, for a LFSR with $l$ memory elements,

- if $l$ is odd, $t = 2^{(l+1)/2} + 1$, and
- if $l$ is even, $t = 2^{(l/2)2} + 1$. 

Figure 6.6 Gold Sequence Generator Circuit
Table 6.2 Gold Sequences - Cross Correlation Property

<table>
<thead>
<tr>
<th>$l$</th>
<th>$m = 2^l-1$</th>
<th>Number of Gold sequence</th>
<th>Max. cross correlation of Gold sequence Normalized</th>
<th>$t$ cross-correlation of Gold sequence</th>
<th>$t/(2^l-1)$ cross-correlation of Gold sequence, Normalized</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>7</td>
<td>2</td>
<td>0.71</td>
<td>5</td>
<td>0.71</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>2</td>
<td>0.60</td>
<td>9</td>
<td>0.60</td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>6</td>
<td>0.35</td>
<td>9</td>
<td>0.29</td>
</tr>
<tr>
<td>6</td>
<td>63</td>
<td>6</td>
<td>0.36</td>
<td>17</td>
<td>0.27</td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>18</td>
<td>0.32</td>
<td>17</td>
<td>0.13</td>
</tr>
<tr>
<td>8</td>
<td>255</td>
<td>16</td>
<td>0.37</td>
<td>33</td>
<td>0.13</td>
</tr>
<tr>
<td>10</td>
<td>1023</td>
<td>60</td>
<td>0.37</td>
<td>65</td>
<td>0.06</td>
</tr>
</tbody>
</table>

Thus, a Gold sequence formally is an arbitrary phase of a sequence in the set $G(u,v)$ defined by

$$G(u,v) = \{u,v,u \ast v, u \ast T v, u \ast T^2 v, U \ast T^{(N-1)} v\}$$

$T^k$ denotes the operator which shifts vectors cyclically to the left by $k$ places, $\ast$ is the exclusive OR operator and $u,v$ are $m$-sequences of period generated by different primitive binary polynomials.

It is well known that the "partial cross-correlation" values can be altered by changing the phases of the code sequences. In theory, then, it is possible to find optimal phases which minimize the interference in the desired data signal. However, for $K$ users each employing a sequence of period $N$, there are a total of $N K$ different sets of sequence phases possible. For a
realistic system, e.g., direct computation becomes intractable. Even when
direct computation is performed, the reduction in interference of the optimal
set of phases over the worst set of phases is 30%.

Finally, the Orthogonal Multiplier multiplies the outputs of
Orthogonal multiplexer with Orthogonal parity vector matrix ‘b’ as shown
below,

\[
S = b \begin{bmatrix}
    c_i \\
    c_j \\
    c_k \\
    c_l
\end{bmatrix} = \begin{bmatrix}
    i^* j^* k^* l^*
\end{bmatrix} \begin{bmatrix}
    c_i \\
    c_j \\
    c_k \\
    c_l
\end{bmatrix} = i^*.c_i + j^*.c_j + k^*.c_k + l^*.c_l \quad (6.3)
\]

where, \(c\) represents the Walsh-Hadamard matrix, ‘s’ represents final encoded
sequence, and ‘b’represents each individual path of sub branch.

### 6.3.6 Design Perspective Example

Multi code generator input is \((1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1)\)\(\Rightarrow\)7 bits

Gold sequence is \(g=(1 \ 1 \ 0 \ 0 \ 1 \ 0 \ 1)\)\(\Rightarrow\)7 bits

Each multi code generator input is multiplied with corresponding
gold sequence is as \((1*1,1*1,0*0,1*0,0*1,1*0,1*1)\)

\[
= \ (1,1,0,0,0,0,1) \Rightarrow7 \text{ bits noted as}
\]

\[
= \ (i^*,i0,j^*,j0,k^*,k1,k0)
\]
Therefore,

\[ i^* = 1; \hspace{0.5cm} i_1 = 1; \hspace{0.5cm} i_0 = 0 \]

\[ j^* = 0; \hspace{0.5cm} j_1 = 0; \hspace{0.5cm} j_0 = 0 \]

\[ k^* = 1; \hspace{0.5cm} k_1 = 0; \hspace{0.5cm} k_0 = 0 \]

The input to the constant envelope encoder is same as the input to the multi code generator is \(a=(1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1)\Rightarrow 7\) bits. The output of constant envelope encoder is as follows

\[ L^* = I^* \wedge J^* \wedge k^*; \hspace{0.5cm} = 1 \]

\[ L_1 = I_1 \wedge J_1 \wedge k_1; \hspace{0.5cm} = 1 \]

\[ L_0 = I_0 \wedge J_0 \wedge k_0; \hspace{0.5cm} = 0 \]

The input to the orthogonal parity vector matrix is \(i^*, j^*, k^*, l^*\) which produces output as \(b=(1 \ 0 \ 1 \ 1)\).

**THE ORTHOGONAL MULTIPLEXER I OUTPUT IS**

\[ i = (0,0,i_1,i_0) \]

\[ = (0,0,1,0) \]

**THE ORTHOGONAL MULTIPLEXER JOUTPUT IS**

\[ j = (0,1,j_1,j_0) \]

\[ = (0,1,0,0) \]

**THE ORTHOGONAL MULTIPLEXER K OUTPUT IS**

\[ k = (1,0,k_1,k_0) \]

\[ = (1,0,0,0) \]

**THE ORTHOGONAL MULTIPLEXER L OUTPUT IS**

\[ l = (1,1,l_1,l_0) \]

\[ = (1,1,1,0) \]
\begin{align*}
S_1 &= b^* I = 1 \\
S_2 &= b^* j = 0 \\
S_3 &= b^* k = 1 \\
S_4 &= b^* L = 0
\end{align*}

Therefore,

\[ S = (s_1, s_2, s_3, s_4) = (1, 0, 1, 0). \]

This is the overall output.

The orthogonal decoder consists of a decision device, correlator banks, orthogonal demultiplexer and Unicode generator as shown in Figure 6.7.

The Decision Device receives the bits and produces an output 1 when the input is greater than 0 and produces an output 0 when the input is lesser than 0. The correlator banks perform the operation of orthogonal multiplication.

The Walsh Code is a mathematical code, otherwise known as the Walsh-Hadamard Code. It is an error-correcting code used to define individual communication channels. The code is calculated by using the Walsh Function.
**Figure 6.7** Multi-orthogonal Decoder Block Diagram

The Unicode generator consists of two blocks, namely, Parallel to Serial Converter and gold sequence despreader which works conversely to the gold sequence generator and is explained in Figure 6.8. The parallel to serial converter converts the data bits into a number of branches according to the length of gold sequence.
6.3.7 Operation of Proposed Link

In this section, the working of the proposed on-chip link is described in detail. EAAK consists of three major parts, namely, ACK, SecureACK (S-ACK), and Misbehaviour Report Authentication (MRA).

The proposed scheme, both the source node and the destination node of a communication link are not malicious, and consider every link between the nodes to be bidirectional.
Our proposed serial on-chip link deals with the problems of collision and limited power transmission as described below. Consider a link in the case of receiver side collisions as shown in Figure 6.9. After node A sends Packet1 to node B, it detects if node B forwarded this packet to node C, at the same time, node X is forwarding Packet2 to node C. In such situations, node A listens to node B whether it has successfully forwarded Packet1 to node C but failed to detect that node C did not receive this packet due to a collision between Packet1 and Packet2 at node C.

In the case of limited transmission power, as shown in Figure 6.10, with the purpose of saving its own battery power, node B purposely limits its transmission power so that it is strong enough to be overheard by node A but not strong enough to be received by node C.

For false misbehaviour report, although node A successfully overheard that node B forwarded Packet 1 to node C, node A still reported node B as misbehaving, as shown in Figure 6.11.

Figure 6.9 Receiver Collision: Both Nodes B and X are Trying to Send Packet 1 and Packet 2, Respectively, to Node C at the Same Time
ACK

ACK is basically an end-to-end acknowledgment scheme. It acts as a part of the hybrid scheme in our proposed scheme, aiming to reduce network overhead when no network misbehavior is detected. In Figure 6.12, in ACK mode, node $s$ first sends out an ACK datagram to the device $d_1$ to the destination node $d$. If all the intermediate nodes along the route between $s$ and $d$ are cooperative and node $d$ successfully receives $P_{ad_1}$, node $d$ is required to send back an ACK acknowledgment packet $P_{ak_1}$ along the same route but in reverse order. Within a predefined time period, if node $s$ receives $P_{ak_1}$, then the
packet transmission from node S to node D is successful. Otherwise, node S will switch to R-ACK mode by sending out an R-ACK data packet to detect themisbehaving nodesin the route.

Figure 6.12 ACK Scheme: The Destination Node is Required to Send Back an Acknowledgment Packet to the Source Node when it Receives a New Packet

R-ACK

The R-ACK scheme is an improved version of the TWOACK scheme proposed by Liu et al. The principle is to test every three consecutive nodes working in a group to detect misbehaving nodes. For every three consecutive nodes in the route, the third node is required to send an S-ACK acknowledgment packet to the first node. R-ACK mode detects the misbehaving nodes in the presence of collision or limited transmission power and the process flow is shown in Figure 6.12.

In R-ACK mode, the three consecutive nodes (i.e., F1, F2, and F3) working in a group to detect misbehaving nodes in the network. Node F1 first sends out R-ACK data packet $P_{sad1}$ to node F2. Then, node F2 forwards this packet to node F3. When node F3 receives $P_{sad1}$, as it is the third node in this three-
nodegroup, nodeF3 is required to send back an R-ACK acknowledgment packet $P_{sak1}$ to nodeF2. NodeF2 forwards $P_{sak1}$ back to nodeF1. If nodeF1 does not receive this acknowledgment packet within a predefined time period, both nodes F2 and F3 are reported as malicious. Moreover, a misbehavior report will be generated by nodeF1 and sent to the source nodes.

Unlike the TWOACK scheme, where the source node immediately trusts the misbehavior report, the proposed method requires the source node to switch to MRA mode and confirm this misbehavior.

### 6.4 EXPERIMENTAL RESULTS

#### Table 6.3 Experimental Results for Power Consumption

<table>
<thead>
<tr>
<th>Components</th>
<th>Power Consumption (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sending Router</td>
<td></td>
</tr>
<tr>
<td>Multi-Orthogonal Encoder</td>
<td></td>
</tr>
<tr>
<td>Multi-Orthogonal Decoder</td>
<td>81</td>
</tr>
<tr>
<td>Receiving Router</td>
<td></td>
</tr>
</tbody>
</table>

The sending router, Multi Orthogonal Encoder, Multi Orthogonal Decoder, and Receiving router Power consumption were given above in Table 6.3.

The sending router Multi Orthogonal Encoder area achieved for 156 slices and Receiving Router Multi Orthogonal Decoder area achieved for 78 slices the experimental result were given Table 6.4.
Table 6.4 Requirements of Slices

<table>
<thead>
<tr>
<th>Components</th>
<th>Slices Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sending Router with Multi-Orthogonal Encoder</td>
<td>156 slices</td>
</tr>
<tr>
<td>Receiving Router with Multi-Orthogonal Decoder</td>
<td>78 slices</td>
</tr>
</tbody>
</table>

![Bar Chart]

**Figure 6.13** Requirements of Slices for Sending Router with Multi-Orthogonal Encoder Receiving Router with Multi-Orthogonal Decoder

The Sending Router with Multi-Orthogonal Encoder Receiving Router with Multi-Orthogonal Decoder slice requirement figure 6.13 were given above.
6.5 SYNTHESIS RESULTS FOR 4-PORT ROUTER

6.5.1 RTL Schematic View 1 for 4 Port

The four port routing algorithm RTL schematic view simulation results were shown in figure 6.14.

Figure 6.14 RTL Schematic view 1 of 4-Port Router
6.5.2 RTL Schematic View2 for 4 Port

The four port routing algorithm RTL schematic view2 simulation results were shown in figure 6.15.

Figure 6.15 RTL Schematic view 2 of 4-Port Router
6.5.3 RTL Layer View for 4 Port

The four port routing algorithm RTL layer view simulation results were shown in figure 6.16.

Figure 6.16 RTL Layer view1
6.5.4 RTL Layer View 2 for 4 Port

The four port routing algorithm RTL layer view 2 simulation results were shown in figure 6.17.

Figure 6.17 RTLLayer view2
6.5.5 RTL Layer View 3 for 4 Port

The four port routing algorithm RTL layer view 3 simulation results were shown in figure 6.18.

Figure 6.18 RTL Layer view3
6.5.6 RTL Layer View 4 for 4 Port

The four port routing algorithm RTL layer view 4 simulation results were shown in figure 6.19.

Figure 6.19 RTLLayer view 4
6.5.7 RTL Layer View 5 for 4 Port

The four port routing algorithm RTL layer view 5 simulation results were shown in figure 6.20.

Figure 6.20 RTL Layer view5
6.5.8 RTL Layer View 6 for 4 Port

The four port routing algorithm RTL layer view 6 simulation results were shown in figure 6.21.

Figure 6.21 RTL Layer view 6
6.5.9 RTL Layer View 7 for 4 Port

The four port routing algorithm RTL layer view 7 simulation results were shown in figure 6.22.
6.5.10 Technology Schematic View 1

The four port routing algorithm technology schematic view 1 simulation results were shown in figure 6.23.
6.5.11 Technology Schematic View2

The four port routing algorithm technology schematic view2 simulation results were shown in figure 6.24.

Figure 6.24 Technology Schematic view2
6.5.12 Technology Schematic View3

The four port routing algorithm RTL schematic view3 simulation results were shown in figure 6.25.

Figure 6.25 Technology Schematic view3
HDL SYNTHESIS REPORT

Macro Statistics

# Latches : 11
  1-bit latch : 1
  10-bit latch : 2
  36-bit latch : 6
  5-bit latch : 2

# Comparators : 7
  36-bit comparator not equal : 3
  5-bit comparator greater : 2
  5-bit comparator less : 2

Device utilization summary

Selected Device : 3s500epq208-4

Number of Slices : 160 out of 4656 3%
Number of Slice Flip Flops : 247 out of 9312 2%
Number of 4 input LUTs : 206 out of 9312 2%
Number of IOs : 147
Number of bonded IOBs : 147 out of 158 93%
Number of GCLKs : 2 out of 24 8%
6.6 SYNTHESIS RESULTS FOR 8-PORT

6.6.1 RTL Schematic View1

The eight port routing algorithm RTL schematic view1 simulation results were shown in figure 6.26.

![RTL Schematic view1 of 8-Port Router](image)

Figure 6.26 RTL Schematic view1 of 8-Port Router
6.6.2 RTL Schematic View 2

The eight port routing algorithm RTL schematic view 2 simulation results were shown in figure 6.27.

Figure 6.27 RTL Schematic view 2 of 8-Port Router
6.6.3 Technology Schematic View

The eight port routing algorithm technology schematic view simulation results were shown in figure 6.28.

Figure 6.28 Technology Schematic view of 8-Port Router
ADVANCED HDL SYNTHESIS REPORT

Macro Statistics

# Latches : 14
10-bit latch : 2
36-bit latch : 10
5-bit latch : 2

# Comparators : 6
36-bit comparator not equal : 3
5-bit comparator greater : 1
5-bit comparator less : 2

Device utilization summary

Selected Device : 3s500epq208-4

Number of Slices : 72 out of 4656 1%
Number of Slice Flip Flops : 102 out of 9312 1%
Number of 4 input LUTs : 40 out of 9312 0%
Number of IOs : 291
Number Of Bonded IOBs : 291 out of 158 184% (*)
IOB Flip Flops : 144
Number of GCLKs : 5 out of 24 20%
6.7 SUMMARY

Semi-seril link is designed using high speed serialization/deserialization and multi-orthogonal encoding techniques. The link also employs acknowledgement scheme to maintain the high speed data intake from the serializer. The proposed semi-seril link was analysed.

The results show that the proposed semi-seril link dissipates the lowest energy per bit. Data is sent from source router to input channel of the communication node through serializer and encoder, and then it is transmitted to the corresponding output router. The proposed serial on-chip link deals with the problems of collision and limited power transmission. Ethiopia Nigussie et al (2012) has designed semi serial on-chip link for router using dual rail encoder circuit. The proposed multi orthogonal encoding technique which consumes power consumption about 73mW. Apart from that the positive acknowledgement in the reference, negative acknowledgements are included which avoids traffic congestion.

The Sending Router with Multi-Orthogonal Encoder consumes 156 slices. The Receiving Router with Multi-Orthogonal Decoder consumes 78 slices. The Sending Router with Multi-Orthogonal and Receiving Router with Multi-Orthogonal Decoder consumes 81mW Power Consumption.

6.6.1 Contribution

- semi-seril link is designed using high speed serialization/deserialization and multi-orthogonal encoding techniques.
• The proposed methodology is a low power and high performance serial on-chip link which includes Sending and receiving routers, serializer, deserializer, multi-orthogonal encoder and decoder.

• The Sending Router with Multi-Orthogonal Encoder consumes 156 slices.

• The Receiving Router with Multi-Orthogonal Decoder consumes 78 slices.

• The Sending Router with Multi-Orthogonal and Receiving Router with Multi-Orthogonal Decoder consumes 81mW Power Consumption.