CHAPTER 4

ADAPTIVE AND RECONFIGURABLE ROUTING

4.1 INTRODUCTION

As discussed in chapter 3, invoking one of the many protocols from the protocol suite as per the throughput can extend the benefits of DRR. Hence, the need for multiprotocol reconfigurable MANET node has become an interesting area of research. This work presents the basic functionalities of the programmable Routing Section (RS) in the MANET node employing Adaptive and Reconfigurable Routing (ARR) technique. Normally, it is assumed that a MANET node transmits data packets by invoking a specific routing protocol permanently embedded in the routing section of the node. Route finding procedure takes place based on this routing protocol.

Considering a situation as shown in Figure 3.9 in which DSR alone is embedded in the routing section of the nodes, based on the observations from the previous work, the network would result in an average throughput of 2616 bps. As opposed, if AODV replaces DSR for the node speed from 1 m/s to 4 m/s and from 10 m/s to 12 m/s, using switching between AODV and DSR, it would result in throughput increase to 3611 bps. Therefore, for the network to result in higher throughput of 3680 bps, it is indispensable that all the nodes do operate either on AODV or DSR protocol interchangeably in these two situations. Hence, if every node is embedded with two or more protocols along with a switching algorithm, then it is possible to extract
maximum throughput under all scenarios by effectively toggling among the protocols with the succor of the switching mechanism.

Samar et al. (2004) advocated that to effectively support communications in a dynamic environment, the routing framework has to be adaptable to the spatial and temporal changes such as traffic and mobility patterns. Therefore, adaptability plays an important role. Also, a node cannot be designed to have many protocols in hardware, which will occupy more space and above all it is not feasible in MANET as nodes are expected to be portable, mobile and battery operative. As a standby, Reconfigurable Field Programmable Gate Array (FPGA) can be relied on.

Saeed et al. (2008) introduced a novel intelligent routing protocol selector, which senses the network performance by using artificial intelligence techniques for modeling and optimization. By combining existing MANET functionality and network context (number of nodes, mobility), the optimal routing protocol can be selected among AODV, DSR, and OLSR. The performance metrics adopted during the comparative analysis are delay, delivery rate and load rate. The decision is made based on the best performance (fewer loads, less delay, higher delivery rate) for each protocol in the same network context.

Jun and Julien (2007) proposed a design tool that automates the evaluation process by controlling environmental and usage properties of the intended deployment. By comparing the environmental and application requirement with extracted protocol behavior model, protocol selection is made. In the later work, the performance metrics considered for the protocol selection process among DSDV, DSR and AODV were average throughput, packet delivery ratio and average end-to-end delay. According to Aggelou (2005), there are two main approaches to allow wireless communication
transceivers (nodes) to operate in a multiparametric, continuously changing environment. The two approaches are:

1. Reconfigurable and adaptive techniques for adjusting the structure and parameters of the transceivers to allow them to demonstrate the best performance in a variety of situations.

2. Robust techniques, which can demonstrate reasonable (required) performance in a variety of the configured situations.

From the works of Saeed et al. (2008), Jun and Julien (2007) and Aggelou (2005), it is learnt that it is wise to use different routing protocols for different situations. Considering all these studies reported in the literature, sticking and adhering to the first suggestion by Aggelou (2005) for the best performance in a variety of situations, ARR is taken up for research. The design is based on the following:

1. Routing section in the MANET node uses maximum number of routing protocols so that ripple in the throughput performance is considerably reduced.

2. Also the node is of a reconfigurable type so that the physical size of the node becomes small.

3. Further, the proposed ARR technique works with high speed of operation due to hardware (Kachris and Vassiliadis 2006).

The proposed routing technique attempts to improve the characteristics of the network to provide higher and consistent throughput by implementing four routing protocols (DSDV, AODV, DSR, and TORA) in a
reconfigurable FPGA. The choice of the routing protocols was limited to four since configuration of many a number of protocols will increase the space and decoding complexity. The choice of the protocols in this thesis consists of one proactive protocol and three reactive protocols. All of these protocols belong to the group, which has minimum performance to maximum performance as shown in Table 2.2. Therefore we decided to include only these four protocols to model an adaptive routing node to maintain higher and consistent throughput.

Reconfigurability means the ability of a device to change its internal structure, functionality, and behavior, either on command, or autonomously. There are various reconfigurability classes, namely, static configuration that is performed while the device is off line. Dynamic configuration is performed while device is on line, “on the fly”. Next, the device itself performs self-reconfiguration autonomously. Therefore, this proposed hardware routing node configures different protocols according to the environmental condition using an ATC algorithm. The architecture using FPGA offers the advantage of achieving more functionality in less area without execution time violation. Further usage of FPGA in routing section of MANET node will result in performance improvement in terms of speed in comparison with other systems.

4.2 MODELING OF ARR NODE

To overcome the issues addressed in the previous research of DRR and AODV-SH, for higher performance and consistency in throughput, switching among different routing protocols in response to the network context has become a viable solution. Also to realize such networks, elements must be able to store, manage and process the context data (e.g. environmental, application, resource information and throughput). This can be executed using three phases like Learn, Decide and Adapt. Figure 4.1
elucidates the basic idea by which learning procedure is carried out in the network context. Depending upon the condition, the parameter is decided to give feedback to the system for adapting.

Figure 4.1 Protocol decision making module (Saeed et al. 2008)

Figure 4.2 Architectural modules for adaptive protocol selection
In Figure 4.2, the adaptability concept is thoroughly explained using the network throughput as the feedback parameter. In this proposed ARR technique, only throughput is used as feedback parameter, which has been demonstrated in the previous contribution of DRR simulation. Usage of four protocols in ARR is due to the fact that the scenarios heavily influence optimum performance of each of the protocol considered (Abolhasan et al. 2004). Therefore, the basic idea of the proposed ARR routing technique is to amend the protocols using reconfigurable FPGA. To implement the technique, it is necessary to study the proposed Automatic Throughput Control (ATC) algorithm, the architecture of MANET node and its router section components, which are explained in the next section.

4.2.1 Architecture of MANET Node

Before implementing ARR technique, the present node’s architecture should be understood as shown below. According to Corson and Macker (1999), MANET nodes are wireless transceivers using antennas which may be omni-directional (broadcast), highly-directional (point-to-point), possibly steerable, or some combination thereof.
The present MANET node has one wireless interface and router architecture with only one protocol embedded in it as shown in the Figure 4.3a. This type of node while forming a network expects the other nodes also to have the same routing protocol embedded in it. In case, if other nodes in the network have different protocol, this might lead to network failure. But in the proposed MANET node architecture as shown in the Figure 4.3b, the formation of network under any environment conditions with any node of similar type (multiprotocol) becomes feasible. At a given point of time, depending on the position of the nodes and their coverage pattern, transmission power levels and co-channel interference levels, a wireless connectivity exists in the form of a random, multihop graph or “ad-hoc” network between the nodes.

This ad-hoc topology may change with time as the nodes move or adjust their transmission and reception parameters. As shown in Figure 4.3b, MANET node should be an integral part of routing section excluding the transmitting and receiving sections meant for routing the data packets through its wireless interface. This routing section or simply the router is shown in the dotted lines forming an add-on card in Figure 4.3b. The router is used to
discover the route for sending its own data to other nodes or to process the data packet from other nodes.

Figure 4.3b Proposed Architecture of MANET node

The router consists of Electrically Erasable Programmable Memory (EEPROM), Reconfiguring Target Section (RTS), Static Read Only Memory (SRAM) and ATC section. The ATC and RTS sections contain one FPGA each. Given the nature of the target application, the reconfigurable component (FPGA) inside RTS or ATC section is organized as an array of reconfigurable cells (RC). RCs have coarse grain granularity. The processor performs scalar operations and controls the operation of RC array. A specialized memory control unit handles data transfer between external memory and RC array and stores data. An expanded version of RTS with a high level block diagram of a one-port router having one input and one output port is illustrated in Figure 4.4. In the following paragraphs the main functions of RTS are presented.
EEPROM is a bank of PROM memories to hold the hardware-implementation details (JTAG) of the routing protocols.

Hardware implementation is fast compared to software implementation since the software contains overhead that is not present in the hardware implementation. Using FPGA based implementation we can achieve parallel operations compared to sequential operation present in normal processor based implementation. Therefore the following steps are carried out to implement the new proposed architecture of MANET node with hardware based routing section.

VHDL modeling is done for four routing protocols, namely, DSDV, AODV, DSR and TORA individually. The simulation and synthesis are done using Xilinx Project Navigator. For VHDL modeling, selection of FPGA device is crucial in the sense, FPGA should have the capability of accommodating maximum number of input output address lines, memories, flip-flops, etc, for any of the four routing protocols at any point of time.

The synthesized code for downloading to FPGA is brought out as JTAG files (e.g. JTAG_DSR, JTAG_AODV, etc). The JTAG files are in the form of stream of bits, which is obtained from the synthesis results. These JTAG files are stored permanently into Programmable and nonvolatile memory ROM and shown as a bank using EEPROM with different selection addresses as S= 00, S= 01, S= 10 and S= 11. Providing enable signals from ATC section, output of EEPROM is fed directly to volatile RTS FPGA.

ATC algorithm that is used for selecting the appropriate protocol is also modeled using VHDL and downloaded into another FPGA called ATC FPGA, which is permanently configured. The complete instructions and procedures of the above implementation are described in the following chapters.
4.2.2 Generic Architecture of Reconfiguring Target Section (RTS)

The detailed block diagram of Generic Architecture of Reconfiguring Target Section is presented in Figure 4.4. The synthesized bit stream file for every protocol is obtained from HDL code. The hardware synthesized code (JTAG file) for four protocols are downloaded permanently into four separate locations of EEPROM. Each memory location will contain the stream of bits as JTAG files of particular routing protocol. When the RTS section is expected to find route on a routing protocol, the corresponding JTAG file is downloaded into RTS FPGA.

![Generic Reconfiguring Target Section (RTS)](image)

Figure 4.4 Generic Reconfiguring Target Section (RTS)

ATC section consisting of ATC algorithm, which is present in another non-volatile FPGA, is responsible for finding which protocol is to be downloaded into RTS FPGA depending on the throughput. If a particular PROM is enabled as per the ATC, the JTAG file of that routing protocol is transferred into RTS FPGA for configuration. During this time, the timer will switch off the data input to RTS FPGA so that JTAG files only will be accepted. Once the RTS FPGA is configured, it starts working on the protocol
to find the route. The configuration of RTS FPGA means that the internal structure of FPGA will be modified such that routing processor, flip-flops, input output devices of FPGA are inter-connected in a different fashion and FPGA will be ready to work for that protocol. The ATC algorithm is a systematic procedure briefed in section 4.3.2 and is responsible for selecting the required routing protocol at that point of time.

4.3 NEED FOR DECISION MAKING ALGORITHM

Normally a MANET node, as a source node when in need of transmitting data packets, invokes a particular routing protocol in the routing section of add-on-card. The routing section works on the particular protocol and collects the route before sending the data packet. The source node does not rely on this routing protocol always because of the changing network conditions. Instead, in the reconfigurable routing section, present protocol at the source will be changed as per the throughput value. Using the algorithm called ATC the change over of protocols is effected. The changing of protocols is done automatically at the source node by comparing the throughput value with the threshold value. It is not sufficient to change the protocol at source alone but the change should be effected in all the nodes in the route path. Hence, auto reconfiguration of routing protocols at intermediate nodes is carried out using information in the reserved field available in the route request packet of the source.

4.3.1 Proposed ATC Algorithm

Though all nodes in MANET invariably should contain ATC algorithm in hardware, the following procedure conforms to the condition that the referred node here is a source. In the beginning, the source node is first reconfigured with DSDV protocol (default). Once the RTS FPGA is reconfigured for any protocol, target section will work as that of router. Then
add-on card along with the routing section will start broadcasting route request through its wireless interface of the node. As it is the source node it instructs the other nodes in the network to reconfigure to the same protocol by inserting a command in the reserved bits field of route request packet. The protocol information bits and their respective routing protocols are shown in Table 4.1.

**Table 4.1 The selection of routing protocol using reserved bits**

<table>
<thead>
<tr>
<th>Reserved information Bits</th>
<th>Routing Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>DSDV</td>
</tr>
<tr>
<td>01</td>
<td>DSR</td>
</tr>
<tr>
<td>10</td>
<td>AODV</td>
</tr>
<tr>
<td>11</td>
<td>TORA</td>
</tr>
</tbody>
</table>

The last section of chapter 3 dealt with the method of changing the routing protocols at run time in a MANET as it provides a better solution in bringing out consistency in throughput. The change of protocol should be done using hardware circuits like FPGA. It was concluded that DRR was able to make changeover of protocols at run time using GloMoSim. But in the hardware, it was understood that some algorithm was essential to execute the change of the routing protocols in RTS FPGA. This is carried out by an algorithm called ATC.

ATC algorithm should work in a round robin manner to select among the popular protocols like DSDV, DSR, AODV and TORA. Further this algorithm should be implemented or stored permanently as software instructions in an FPGA. This ATC FPGA is programmed only once and stored permanently. The following section discusses the various steps
involved in development of that algorithm. ATC FPGA should have the various stages like system clock, acknowledgement counter and system counter, etc. ATC section should be able to use the external FIFO data memory and buffer memory with required control signals. The procedure of ATC algorithm and selection of routing protocols at run time are illustrated using the flow chart in Figure 4.5.
Figure 4.5 Flow chart of ATC algorithm
4.3.2 Specifications of ATC Algorithm

1. In idle state, system’s clock and system counter are reset.

2. Whenever the user is ready to transmit data, source node is switched on.

3. At reset default routing protocol is configured in RTS FPGA. DSDV protocol is proactive [as a default] and moves to reactive protocols in round robin method i.e. DSR, AODV, and TORA.

4. Route to destination is discovered by RREQ packet as per the routing protocol, which is configured in RTS FPGA of source node at that point of time. This flooding of packet to other nodes contains the information in the fields, for which protocol, the intermediate and destination nodes have to be configured. After the neighbor nodes are configured for the protocol, the route is established and the sink node sends back the detailed route in the reply packet. Then the actual data transmission by the source will start.

5. The route request packet sent by the source should contain the fields like the source id, destination id, and sequence number, time to live parameter, the routing protocol to be used, etc, on which the source node requires the route to the destination.

6. Source sends the data packets through the obtained route that are placed in the FIFO for transmission. The sending device has to keep a copy of the frame sent until it receives an ACK. Hence source duplicates all the data packets and places in the SRAM (buffer). Here the FIFO and SRAM buffer memory are of equal capacity.
7. System clock, Data counter and ACK counter are switched on, data counter is incremented whenever the data packet is transmitted from the source. Data are forwarded based on the routing information.

8. As soon as the acknowledgement is received by the source, SRAM buffer memory content of the corresponding packet is deleted.

9. Also, as soon as ACK is received from the destination, the acknowledgement counter is incremented for every receipt of acknowledgement.

10. In the simulation, we have assumed node speed to be 25m/s moving in random fashion in an 800m x 800 m grid with UDP packet rate of 4 Packets/sec of each 256 bytes. This leads to a rate of 4x256x8 bps. In DRR, the throughput reaches to a maximum of 8000 bps. Further, the nodes take 32 sec to cross the entire grid of 800x 800 grid. Here, the node has to wait for 10 sec (80000 bits) before the network condition changes due to mobility and take corrective action. Too high or too low observation time will degrade the network performance. After 10 sec of data transmission, source node should suspend transmission temporarily for throughput observation.

11. After data transmission is suspended, the following procedure is done.

Find the Initial throughput value as number of packets successfully transmitted per unit time, i.e., ACK counter value / 10 sec. This ratio will give throughput in number of packets per second (T0).
12. Automatic Throughput Control is designed with the threshold value of 50% throughput, due to two reasons. (i) When threshold is too low, the throughput due to active protocol can be below the target value for a long time before switching is made to better protocol. (ii) When the threshold is high, the throughput due to active protocol may often fall below the threshold value, which will cause the protocol selector to switch frequently to other protocols and cause more swapping delay reducing the overall throughput. Since, fixing the threshold too high or too low degrades the performance, it is customary to fix, half of the Initial throughput value (T0/2) in the memory (0) for future comparison.

13. Use the same default routing protocol configuration and start transmitting the remaining data packets from FIFO for a duration of 10 sec.

14. Stop transmission of data packets and read T1 from acknowledgement counter.

15. Find out the new throughput value (T1), store it in memory1 and compare with the initial value which is stored in the memory (0) i.e. T0/2 = 0.5T0.

16. Throughput comparison procedure is as follows:

- If the new value (T1) exceeds the initial throughput value (T0/2) [i.e. T1 = T0/2], continue transmission of remaining data packets in the FIFO with the same default protocol used.

- If not [i.e. T1 < T0/2] source node shifts to the DSR protocol which is next in the order using commands from
EEPROM to find the new route. Transfer of JTAG file of DSR to RTS FPGA carries out this work. The transfer is the simple action of enabling EEPROM memory to do so by the control signal from ATC FPGA.

- For condition [i.e. $T_1 < T_0/2$], start finding the route for the destination by flooding RREQ (all the intermediate nodes reorganize and reconfigure their routing strategy to the new protocol). The field of RREQ packets will contain the code for corresponding protocol. At the same time acknowledgement counter value is reset to value 0. This will take a few seconds.

- Once source receives the route to the destination by the present protocol, it will start transmitting the remaining data in FIFO.

- Again find out throughput value [T2] and compare with the memory (0) [T0/2].

- If the present value exceeds [i.e.$T_2 \geq T_0/2$] memory (0), continue data transmission from FIFO with the present routing protocol.

- If not [i.e. $T_2 \leq T_0/2$] change to the other protocol, and that will continue till the last protocol (TORA) comes into effect. Protocols are transferred based on round robin basis from EEPROM to RTS FPGA. In case of more routing protocols to be used for an effective high throughput achievement, the simplest criterion is to switch randomly to any of the protocols based on threshold value and continue. In ATC, any method can be followed to select
between protocols including the one based on Table 2.2 or based on the below procedure. In this work, we considered DSDV (proactive) to be the reference protocol to evaluate the throughput during initial network setting and move on to reactive protocols subsequently like next effective protocol DSR. Further it moves to AODV that is a combination of DSDV and DSR and finally to a more effective multi route protocol TORA.

17. If all the subsequent protocols do not satisfy the throughput condition in memory (0), then, the default protocol DSDV (first one in EEPROM) is again used for transmission of data packets and at the same time memory (0) value is changed to a new value as below.

- Throughput value (memory (0)) i.e. $T_0/2 = 0.5T_0$ which was used till now for comparison is changed to the new value [i.e. $T_0*2/5 = 0.4T_0$] and stored in the same location memory (0) as new value.

- The same procedure is followed for rest of the data packets in FIFO, starting with the default protocol. For every full cyclic change of routing protocols, the memory (0) value is reduced by 10 percent of $T_0$.

18. Sometimes it may be observed that the throughput value becomes zero at some instant of time of the data transmission (i.e. $T_1$ or $T_2$ or ... $T_n = 0$). In this condition, the source stops further transmission of data packets and goes into the idle state and wait for a while to transmit the remaining packets. Getting the throughput value as 0 means the path could not be established by all the protocols and that may be due to link
failure, network segmentation or non-cooperation of intermediate nodes, etc. The duration time, the source node has to wait before it can start a new transmission of remaining packets is calculated as $T_s/2$ where $T_s$ = number of packets already reached the destination/ the time already consumed. This implies that there is a chance of some node in the network to move inside the radio range so that a new route to the destination can be established.

19. Based on the packet acknowledgement numbers, the corresponding data packets are deleted from SRAM buffer. Regarding the previously received packets by the destination, the packets will be buffered till the rest of the packets arrive at the destination so that it can be concatenated and decoded to obtain the original message.

20. At some point of time, still some packets are available for deletion in SRAM buffer memory for want of acknowledgement, and then the packets are copied to FIFO for retransmission.

21. The existing routing configuration is continued for transmission till buffer memory becomes empty.

### 4.4 HARDWARE REALISATION OF ATC

The implementation of ATC algorithm follows the procedure of automatic throughput control algorithm explained in section 4.3. The state modeling of ATC algorithm using 8 finite states is shown in Figures 4.6 and 4.7. It reveals all the basic processes carried out by ATC algorithm in the reconfigurable routing section in the proposed node architecture.
procedure includes configuration of RTS FPGA for a particular protocol, which is one out of the four protocols, namely, DSDV, DSR, AODV and TORA. The ultimate decision by ATC in turn, depends on the threshold throughput value, threshold value computation, computing throughput at each instance and comparison, etc. The hardware implementation of ATC algorithm includes more or less the same procedure used for individual routing protocol implementations. For ATC implementation, coding in VHDL, synthesizing using Xilinx Project Navigator tool and simulation by ModelSIM are systematically carried out.

![Figure 4.6 ATC finite state machine diagram](image-url)
Device utilization summary:

- Selected Device: 3s500epq208-4
- Number of Slices: 31 out of 4656 (<1%)
- Number of Slice Flip Flops: 20 out of 9312 (<1% 0%)
- Number of 4 input LUTs: 54 out of 9312 (<1% 0%)
- Number of IOs: 31
- Number of bonded IOBs: 29 out of 158 (18%)
- IOB Flip Flops: 6
- Number of GCLKs: 1 out of 24 (4%)

Timing Summary:

- Minimum period: 3.763 ns
- Maximum Frequency: 265.745 MHz
Minimum input arrival time before clock : 17.043 ns

Maximum output required time after clock : 4.368 ns

4.5 CONCLUDING REMARKS

This chapter detailed the ARR routing technique using RTS FPGA. The routing technique presents the method of changeover of protocols using reconfigurable FPGA. It further explained the difference between the ‘node or router node’ and ‘router or routing section’ with their architectures and the steps involved in the creation of reconfigurable node with FPGA. The controlling part of the reconfigurability of RTS FPGA in routing section is an important procedure and is based on continuous throughput monitoring. For this purpose an algorithm called ATC is proposed and implemented in hardware (ATC FPGA). Using ATC algorithm, changeover of protocols is effected. ATC algorithm is used to select the appropriate routing protocol using a series of procedures. The changing of protocols is done automatically at the source node by comparison with the throughput value with a threshold value. The hardware realization of ATC algorithm in ATC FPGA in the ATC section is explained in the latter part of the chapter.