TTL Implementation of a CAMB Tree Network Switch [1]

Technical Report 91-25

Hung Khei Huang, Kenneth A. Godbille, Tatsuya Suda

Department of Information and Computer Science

University of California, Irvine

Irvine, CA 92717

[1]This material is based upon work supported by the National Science Foundation under Grant No. NCR-8907909. This research is also in part supported by University of California MICRO program.

Abstract

Packet collisions and their resolution create a performance bottleneck in random-access LANs. A hardware solution to this problem is to use a collision avoidance switch. These switches allow the implementation of random access protocols without the penalty of collisions among packets. An architecture based on collision avoidance is the CAMB (Collision Avoidance Multiple Broadcast) Tree network, where concurrent broadcasts are possible.

The purpose of this paper is to present two implementations for a CAMB Tree switch[*] *. First, a general outline of the CAMB switch is provided. Then, a description of the two implementations is given.*

1. A CAMB Network.

To fully understand the operation of the tree network switch proposed, this section attempts to briefly describe the principle behind a CAMB Tree Network.

As its name implies, a CAMB Tree Network consists of a set of switches arranged in a tree topology (Fig. 1). Data packets enter and exit the network at the leaf switches. (Each leaf switch is connected to a number of network stations or devices.) The packets are routed up the tree until they reach a switch that is the root of the smallest subtree which contains both the source and destination stations. This switch is know as the *least common ancestor* of the source and destination stations. Packets are then broadcast to every station in this subtree. The receiving stations recognize when a packet is meant for them by a destination address which starts each packet.

[*] *A short summary of a CAMB Tree network is given here, but for more information, I suggest the reader consult "Tree LANs with Collision Avoidance: Station and Switch Protocols" by Dr. Tatsuya Suda and Steve Morris [Computer Networks and ISDN Systems. Elsevier Science Publishers B.V., North-Holland, 1989, vol. 17, pp. 101-110].*

1.1 The CAMB Tree Switch.

The CAMB Tree Switch can be understood in terms of three smaller modules (shown in Fig. 2):

- The *Uplink Selector* selects a packet at random from the set of incoming packets.
- The *Address Recognizer* determines if the switch is the *least common ancestor* of the source and destination stations.
  - -If the switch is not the least common ancestor, the Address Recognizer routes the incoming packet up the tree to the parent switch.
  - -Otherwise, the incoming packet is sent to the Downlink Selector, described below. In addition, the Address Recognizer sends the packet up the tree. The reason for sending this packet will be revealed later.
- The *Downlink Selector* broadcasts a packet from either the Address Recognizer, or from the parent switch's downlink. The Downlink Selector gives priority to the parent downlink. Furthermore, if a packet from the parent downlink arrives during a packet transmission from the Address Recognizer, the Downlink Selector will abort the packet transmission from the Address Recognizer in favor of the packet on the parent downlink line. Also, any packets coming from the Address Recognizer during a parent downlink transmission are blocked.

In the description of the Address Recognizer, it was stated that during a packet broadcast the switch will also send the packet up the network tree. The reason for doing this is the protocol which drives the Downlink Selector. When an incoming packet is accepted for broadcast, the Downlink Selector allows the packet to be aborted by a transmission from the parent switch. As a result, it is possible for a switch to be continually forced to abort broadcasts due to other packets accepted by another switch higher in the tree. To resolve this problem, a way must be found to block switches higher in the tree from acquiring new packets during a broadcast.

The solution adopted by the CAMB Network presented here is to insure that the switch has the following properties: 1) the selected packet is always transmitted up the tree, and 2) the packet busies switches higher in the tree so no new packets are accepted by the network.

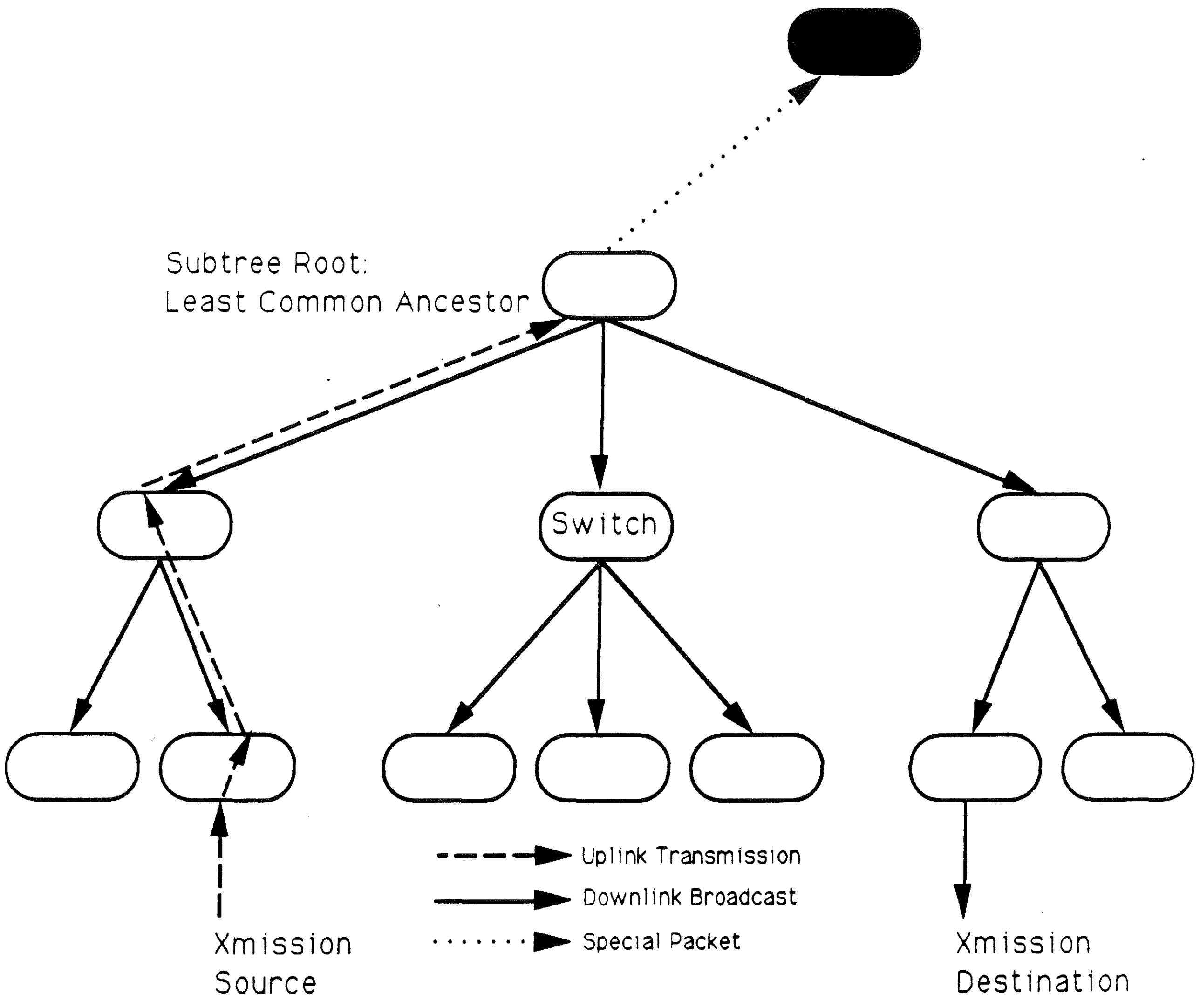


Fig. 1: A Tree Network.

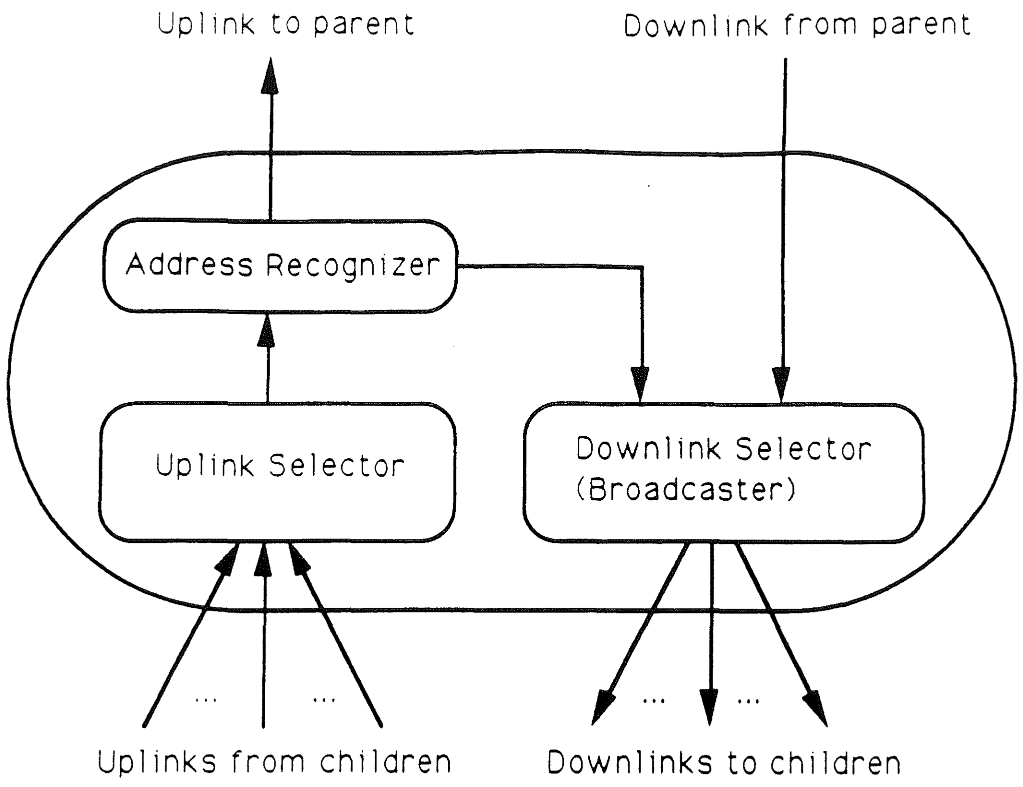


Fig. 2: A CAMB Switch.

2. Implementing the Switch.

This section gives a high level description of a specific implementation of the CAMB switch. Below, a list is given which describes the specific attributes of the proposed switch.

2.1. Switch Interfaces.

Following is a list which describes a specific implementation of the CAMB switch in relation to the general description given earlier (refer to Fig. 3):

- Each switch in this design has four uplinks and four downlinks. This means leaf switches can interface with four devices (stations)-- the entire tree structure is a 4-ary tree.

- Each uplink or downlink can be thought of as having a *control path* and a *data path*. The control path signals the beginning and end of each packet. The data path is responsible for both packet routing and conveying packet information.

- Each switch is connected to its parent switch with one uplink and one downlink.

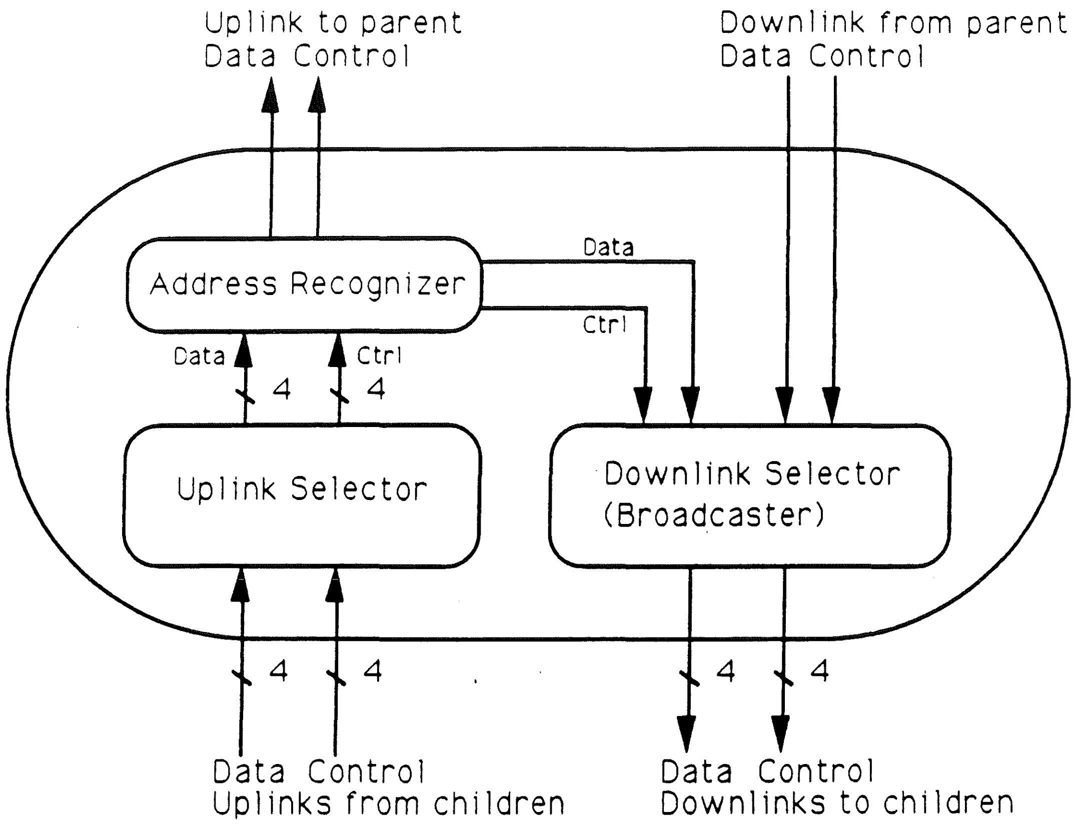


Fig. 3: A TTL CAMB Switch.

3. A Hardware Implementation.

This section describes two implementations of a CAMB Tree Switch at the hardware level. Full schematics of the TTL tree switch are provided.

3.1. Switch Implementation # 1.

This section gives a description of one of the switch implementation presented in this paper. First we give a description of the packet format and the addressing scheme assumed in this implementation. Then we present the block diagram of each one of the sections of the CAMB switch. Finally we describe the signals and the circuit schematic.

3.1.1 Packet Format.

This implementation of the CAMB Tree Network has a packet format with four fields: a *Destination Address Field, Control Bit Field, Packet Size Field* and *Data Field* (Fig. 4).

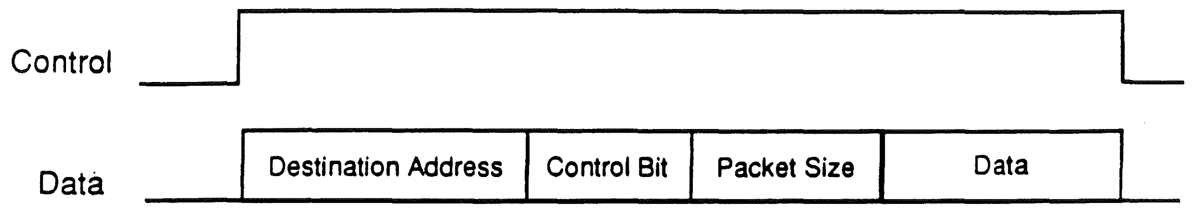


Fig 4. Packet Format

The Destination Address field (8 bits long) indicates the destination address. The next field is the Control Bit (1 bit long). This bit is used to indicate if the packet has reached its proper ancestor. This bit is initially set to 1 by the sending station. It remains 1 until the packet reaches the proper ancestor. When a packet reaches its proper ancestor, this switch resets this bit to 0.

The Packet Size field indicates the number of bits contained in the data field. This information is used by the stations (not by the switches) to determine whether the packet has been aborted. If the Packet Size field is equal to the size of the data field, the packet is successfully transmitted, otherwise, abortion happened. Note that, depending on higher layer protocols, other overhead information such as source address and CRC may be necessary in the packet.

The Control line is high whenever there is a valid packet transmission in the Data line. Otherwise it is low.

3.1.2. Addressing Scheme.

The addressing scheme assumed in this switch design is shown in Fig 5. Stations are ordered in ascending order from left to the right of the tree. Each station has a 2x(M -1) bit long address, where M is the height of the tree. Address are assigned to the switches in the following way. The level i switches have a 2x(i - 1) bit long address. The address of a switch on level i matches with the first 2x(i - 1) bits of the addresses of all the stations in the subtree.

The above numbering of stations and switches makes it easy for a switch to know whether it is the proper ancestor of a packet . A switch on level i checks the 2x(i - 1) leftmost bit of the address field of a packet. If it is equal to its own address, then it is the proper ancestor of the packet. In this case, the packet is sent to both the parent switch and the downlink switches. Otherwise, the packet is only sent to the parent switch.

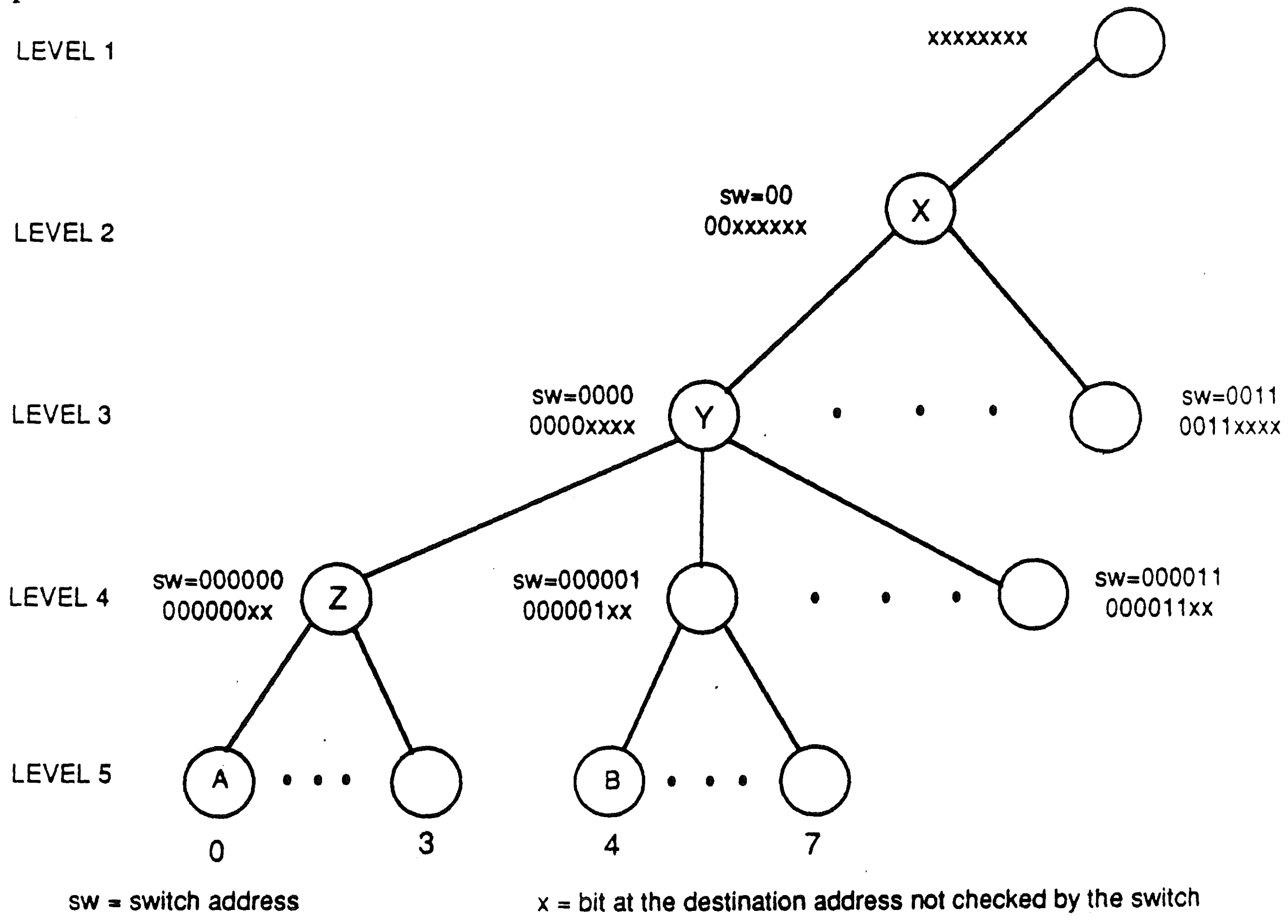


Fig 5. Addressing Scheme

Fig 5 shows a transmission of a packet from station A (address 00000000) to station B (address 00000100) as an example. Station A's immeadiate parent switch, Z (address 000000), checks the destination address of the packet. As the six leftmost bits of the destination address in the packet header is different from the switch address, it sends the packet only to its parent switch (Y). Switch Y (address 0000) then checks the four leftmost bits of the destination address of the packet. This time, as they match with the switch's address, switch Y knows that it is the proper ancestor of the packet. Thus, the packet is broadcast to its children. The packet is also sent to the parent switch (X).

Now, the switch X (address 00) checks the two leftmost bits of the destination address field. They agree with the switch address. Therefore, the switch X knows that it is the proper ancestor of the packet and becomes ready to broadcast the packet to its children. However, if the switch X actually broadcasts the packet, it is broadcast twice (once by switch Y and once by switch X). To avoid this undesirable situation where the same packet is broadcast more than once by different switches, a control bit has been added in the packet header (fig 4). First, a switch checks the control bit in the packet header. If it is 0, a switch knows that the packet has already been broadcast by its proper ancestor, so it will transmit the packet only to the parent switch. Otherwise, it will check the address field to see if it is the proper ancestor of the packet. In the example shown above, the switch X first checks the control bit. As it is 0, the switch does not broadcast the packet.

3.1.3. Switch Structure.

To better understand how the TTL implementation works, the switch has been broken down even further. In Figure 6, the Uplink Selector has been broken down into four parts.

- The *Synchronizer* - which aligns an incoming packet with the switch's internal clock. One important observation: if several switches in the network share the same clock signal, this unit is not required between interconnected switch nodes. Removal of the unit in this case will speed up packet transmission.

- The *Start Recognizer* - which detects the start of a new packet. This unit also allows the Uplink Selector to block all other uplinks while a selected packet is being transmitted.

- The *Priority Resolver* - which selects one packet at random from the set of new packets.

- The *Multiplexer* - which acts as a gatekeeper, allowing only the packet which the Priority Resolver selects through to the Address Recognizer.

Notice that the Start Recognizer and the Priority Resolver are exclusively control path units, i.e., they do not connect to the data path.

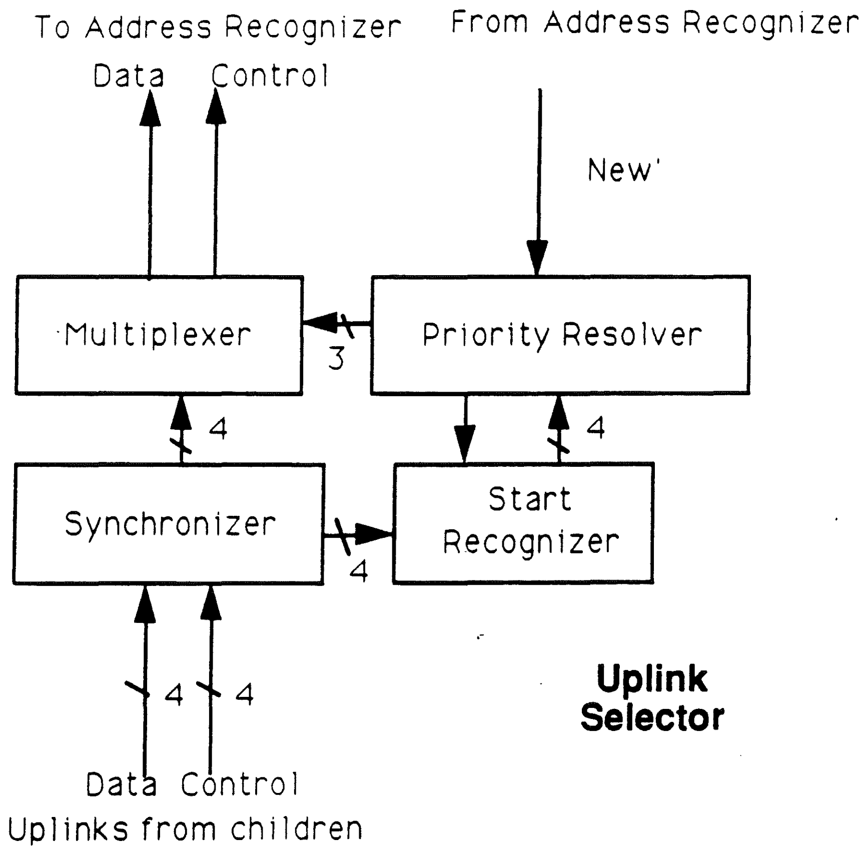


Fig 6. CAMB Switch Uplink Selector

Likewise, the Address Recognizer (fig. 7) can be reduced to its component parts.

- The *Shift Register* - stores the destination address, so it can be checked by the comparator. Note that the shift register always sends the incoming packet to the parent switch.

- The *Comparator* - generates a signal (Cmp) which indicates whether the incoming packet destination address matches with the switch address.

- The *Routing Logic* - generates a signal (Broadcast) which indicates whether the incoming packet will be sent to the Downlink Selector to be broadcast or not. This signal is generated based on three signals: Cmp, Control Bit, Parent Int'.

- The *Reset Control Bit Logic* - resets the control bit in the incoming packet whenever the switch recognizes the it is the proper ancestor of the packet. This indicates that the packet has already been broadcast.

- The *End Recognizer* - which detects the end of a transmitting packet or a transmission interrupt from the parent downlink line. When one of these events happens, the End Recognizer sends a reset signal (New') to the Uplink Selector.

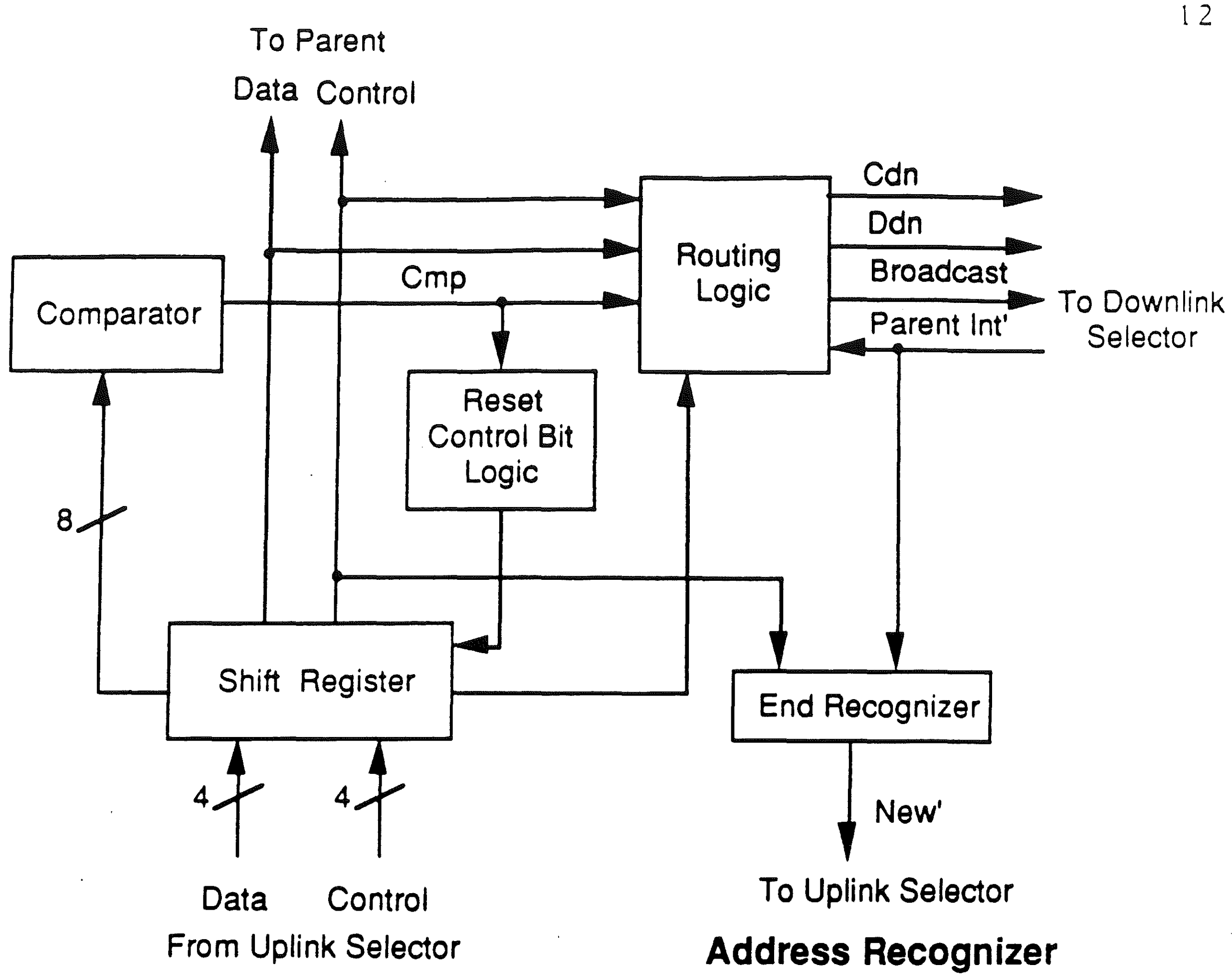


Fig 7. CAMB Switch Address Recognizer

The Downlink Selector (fig. 8) can be reduced to its component parts.

- The *Selector* - selects a packet either from the parent downlink or from the Address Recognizer to be broadcast to all the children switches. Priority is given to the parent downlink. So, if a packet from the parent downlink arrives during a packet transmission from the Address Recognizer, the selector will abort the packet transmission from the Address Recognizer. In addition, any packet from the Address Recognizer will be blocked during a parent downlink transmission.

- The *Flip-flop* - is used to force the control line to go low when abortion happens. So if abortion occurs the control line will go to low, indicating the end of the transmission of the packet from Address Recognizer. A station detects an abortion by checking the packet size indicated in the packet header and the actual size of the packet received.

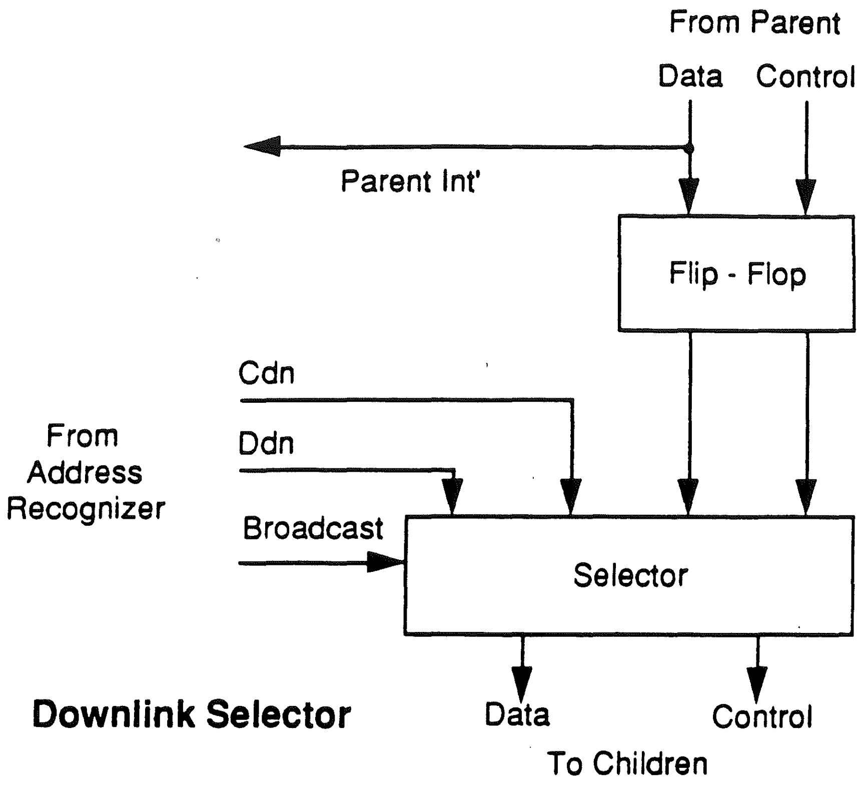


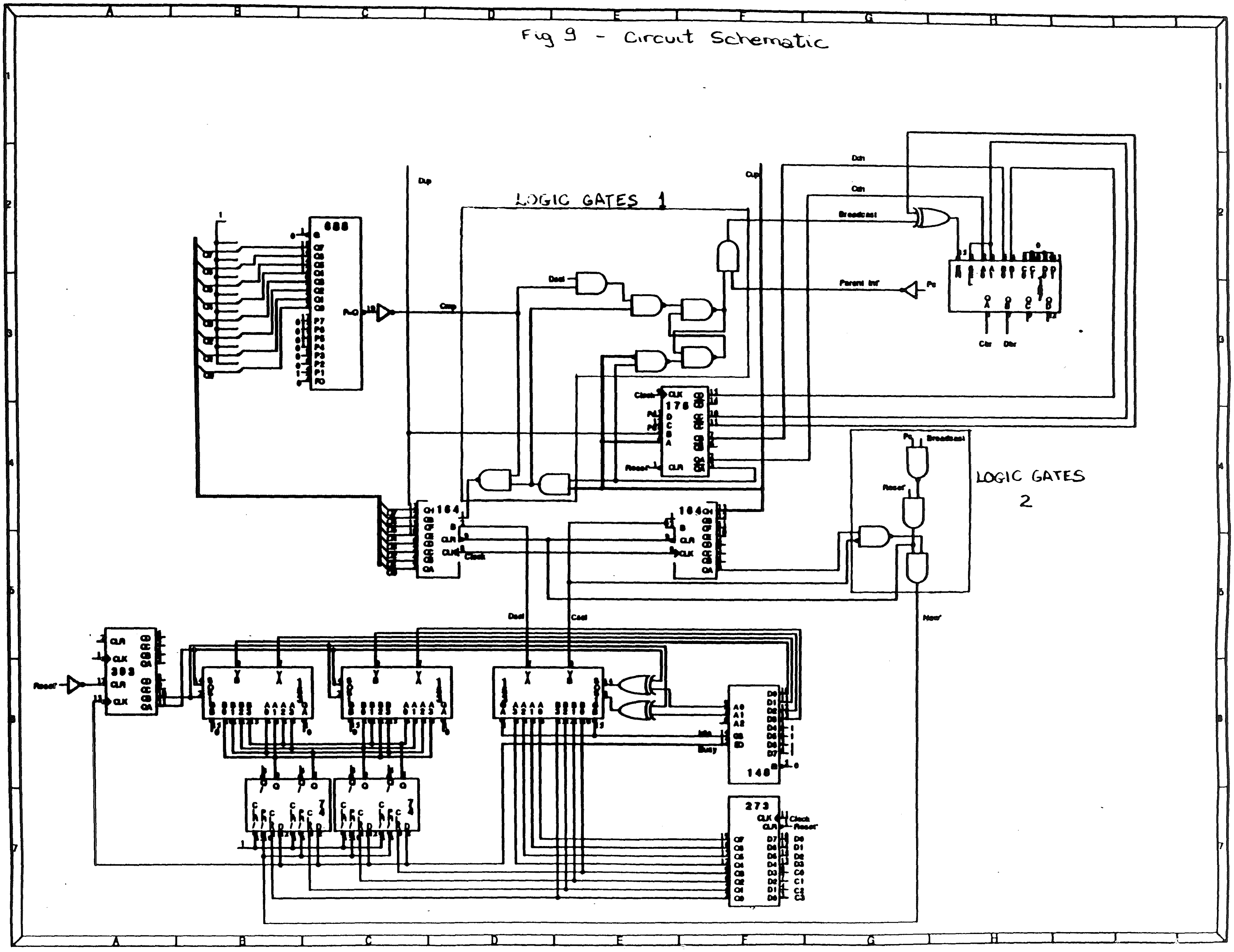
Fig 8. CAMB Switch Downlink Selector

3.1.4. TTL Implementation - The Signals

The next sections describes the implementation of the CAMB Tree Switch at the hardware level. A full schematic of the TTL tree switch is given in Figure 9. An explanation of the the signals shown is given below.

Signal	Purpose
(Inputs)	
0	Ground.
1	Positive power lead.
Clock	A periodic clock signal.
Reset'	Master Reset: it activates the New' signal, aborting any ongoing transmission, and readying the switch to receive a new packet.
D0...D3	Data inputs. An uplink or downlink line consists of one data input, Dx, and one control input, Cx ($1 \leq x \leq 3$).
C0...C3	Control inputs.
Pd	Parent downlink data line.
Pc	Parent downlink control line.
(Internal signals)	
New'	Resets the switch to receive a new packet. This signal is complimented, i.e., the switch is reset when New' is 0. New' resets the switch if one of the following events occurs: 1) end of packet, 2) the parent downlink interrupts or blocks a packet from the Address Recognizer, or 3) a Reset signal is given.
Idle	This signal is high when no packet is selected.
Busy	This signal is high when a packet is selected.
Dsel	Data line of the packet selected.
Csel	Control line of the packet selected.
Cmp	It is 1 if the switch is the proper ancestor of the packet (destination address is equal to the switch address and control bit is equal to 1). Otherwise is equal to 0.
Broadcast	This is equal to 1 when a packet from the Address Recognizer is being broadcast.
Parent Int'	Indicates that a packet from the parent downlink is being broadcast.
Ddn, Cdn	These two signals carry packets to the Downlink Selector.
(Outputs)	
Dup	This signal is the parent switch data uplink. It will become some Dx on the input to the parent of this switch.
Cup	This signal is the parent switch control uplink.
Dbr	This is the data downlink of the switch which carries a broadcasted packet to 1-4 children switches or devices.
Cbr	This is the control downlink of the switch.

Fig 9 - Circuit Schematic



3.1.5. A TTL Implementation - The Circuit

In figure 9, the entire tree switch node is drawn. This section will give a brief rundown of the different parts of the implementation.

On the lower half of the page is the Uplink Selector:

- The 273 at the bottom of the page is an eight-bit latch, and it receives the input from the switch uplinks. The latch functions as the Synchronizer in the Uplink Selector. The resister array just before the latch pulls down to zero (the idle state) any unconnected inputs.

- The two 74 chips (D flip-flops) directly above the 273 function as the Start Detector. They detect the rising edge of each control line at the start of a packet transmission.

- The Priority Encoder 148, the XOR gates, the two Multiplexers 153 and the Counter 393 function as the Priority Resolver. The XOR gates, the two Multiplexers 153 and the Counter 393 provide a rotating priority by shifting which line has priority at any given time. The 148 outputs the number of the selected input. This line is converted by the XOR gates to select the appropriate active incoming uplink.

The upper half of the page contains the Address Recognizer and the Downlink Selector:

- The Comparator 688 compares the destination address with the switch address. The switch address is determined by a set of DPDT switches.

- The Shift Registers 164 store the destination address.

- The 175's flip-flops A and B are used to store the packet for one cycle while the address recognizer is being analyzed by the 688.

- The group of logic gates 1 form the Routing Logic and the Reset Control Bit Logic.

- The group of logic 2 forms the End Recognizer.

- The Multiplexer 157 acts as a selector.

- The *Deglitcher/Resynchronizer* realigns the outgoing signal with the clock.

- The *End Recognizer* detects the end of a transmitting packet and resets the Uplink Selector. The End Recognizer is also responsible for detecting an interrupting transmission from the parent downlink line. (Again: a transmission on the parent downlink will block or abort a broadcast transmission from the Address Recognizer.)

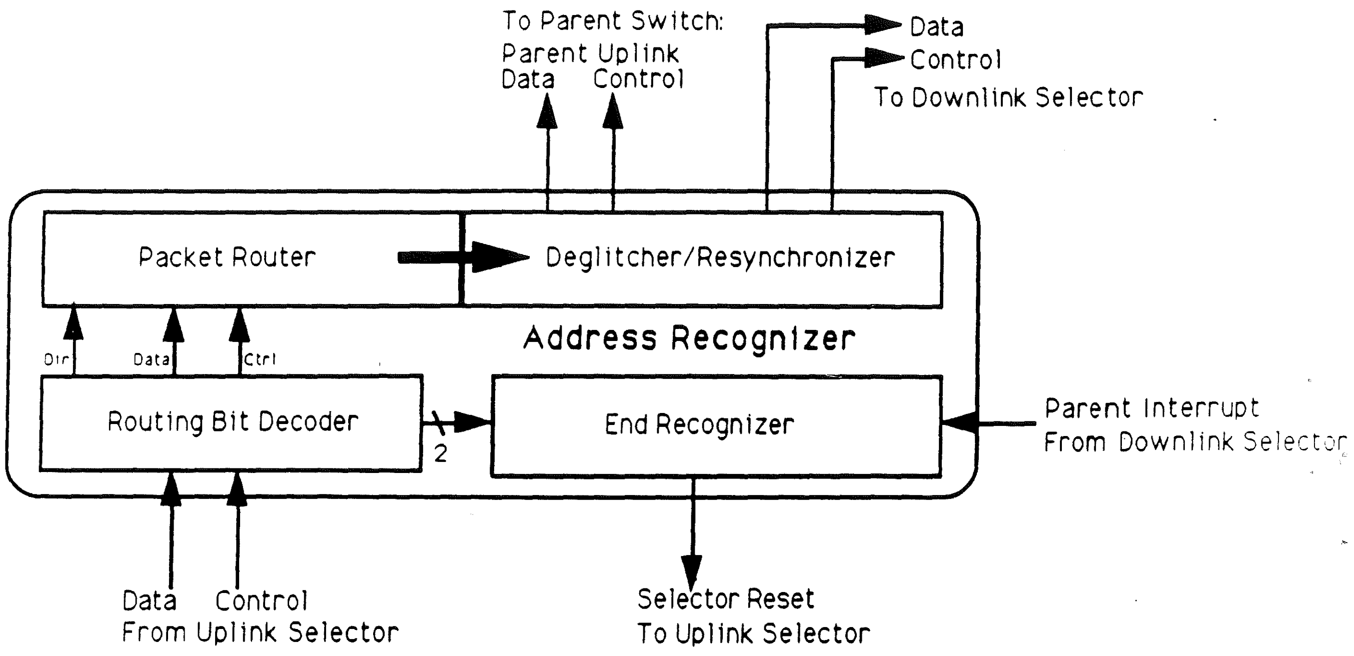


Fig. 11: The Address Recognizer.

It is not necessary to fragment the last switch component, the Downlink Selector, into smaller pieces (fig. 12). The Downlink Selector's main function is to choose a packet from either the Address Recognizer or from the parent downlink, and broadcast that packet to all child switches. If a packet from the parent downlink arrives during a packet transmission from the Address Recognizer, the Downlink Selector will abort the packet transmission from the Address Recognizer. In addition, any packets from the Address Recognizer will be blocked during a parent downlink transmission.

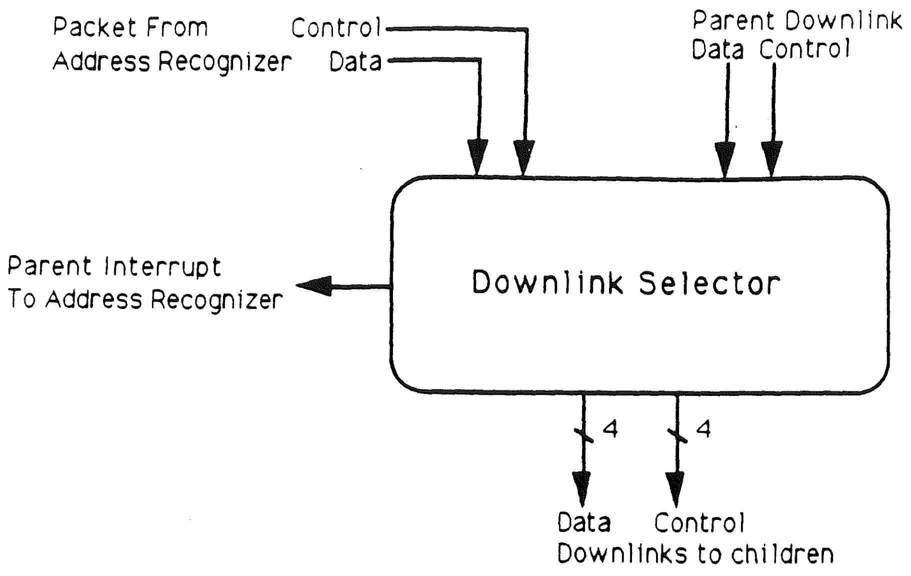


Fig. 12: The Downlink Selector.

3.2.2. TTL implementation.

This section describes an implementation of a CAMB Tree Switch at the hardware level. A full schematic of the TTL tree switch is given in Figure 13. An explanation of the signals shown is given below.

One of the main philosophies behind this TTL implementation of a CAMB Tree Switch, is to increase the throughput of the network by minimizing the time each transmission takes in the network. To accomplish this, each packet undergoes a short pre-processing step at its source station. How this processing takes place-- weather the processing occurs at the hardware interface, or at the software level-- will determine the final form of the addressing scheme and pre-processing algorithm used. For completeness, one possible addressing scheme and pre-processing algorithm is presented below.

Signal	Purpose
(Inputs)	
0	Ground.
1	Positive power lead.
Clock	A periodic clock signal.
Switch Reset	Switch Reset: it activates the New' signal, aborting any ongoing transmission, and readying the switch to receive a new packet.
D0...D3	Data inputs. An uplink line consists of one data input, Dx, and one control input, Cx ($0 \leq x \leq 3$).
C0...C3	Control inputs.
Pd	Parent downlink data line.
Pc	Parent downlink control line.
(Internal signals)	
New'	This signal, generated by the End Recognizer, resets the switch to receive a new packet. The signal is complimented, i.e., the switch is reset when New' is 0. New' resets the switch if one of the following events occurs: 1) end of packet, 2) the parent downlink interrupts or blocks a packet from the Address Recognizer, or 3) a Switch Reset signal is given.
Idle	This signal is high when no packet is selected.
Busy	This signal is high when a packet is selected.
Mux0, Mux1	These bits describe the uplink currently selected by the Uplink Selector. Mux0 is the LSB.
Dsel	Data line of the packet selected.
Csel	Control line of the packet selected.
Dir	Goes high when the switch is the least common ancestor of both the source and destination device, and low otherwise.
Cout	This is the new control signal for a packet exiting the switch.
Pidle	Goes high when the parent downlink line is not transmitting a packet.
Ddn, Cdn	These two signals carry packets from the Address Recognizer to the Downlink Selector.
(Outputs)	
Dup	This signal is the parent switch data uplink. It will become some Dx on the input to the parent of this switch ($0 \leq x \leq 3$).
Cup	This signal is the parent switch control uplink.
Dbr	This is the data downlink of the switch which broadcasts a packet to 1-4 children switches or devices (stations).
Cbr	This is the control downlink of the switch.

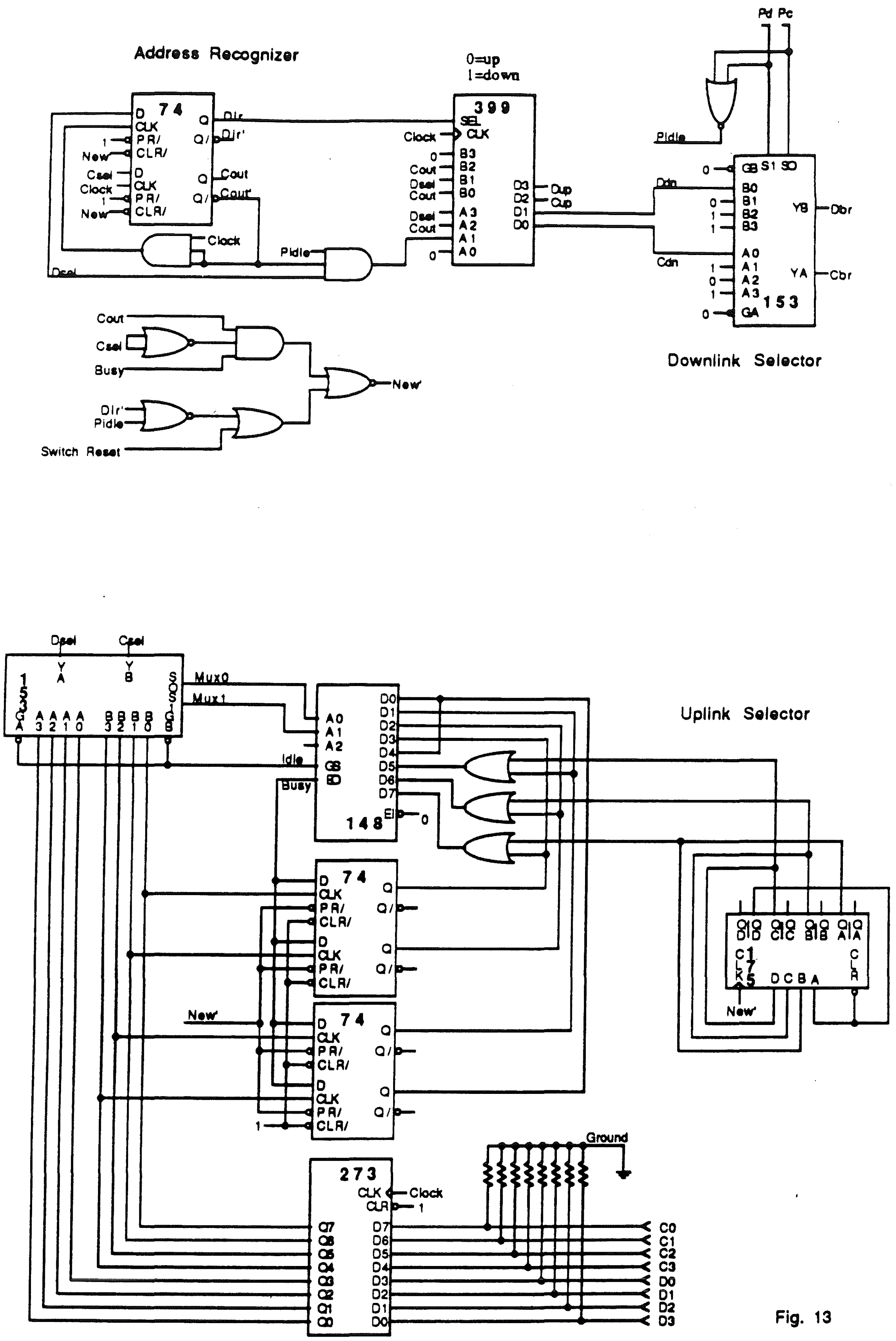

Address Recognizer
74
D
CLK
PR/
CLR/
Q
Q/
Dir
Dir'
Cout
Cout'
New
Csel
Clock
Dsel
Pidle
0=up
1=down
399
SEL
CLK
B3
B2
B1
B0
A3
A2
A1
A0
D3
D2
D1
D0
Dup
Cup
Pd
Pc
Pidle
Ddn
Cdn
GB
S1
S0
YB
Dbr
YA
Cbr
GA
153
Downlink Selector
Cout
Csel
Busy
New'
Dir'
Pidle
Switch Reset
Dsel
Csel
153
Mux0
Mux1
148
A0
A1
A2
GS
EO
EI
Idle
Busy
Uplink Selector
74
New'
175
CLK
CLR
New'
Ground
273
CLK
CLR
Clock
Q7
Q6
Q5
Q4
Q3
Q2
Q1
Q0
D7
D6
D5
D4
D3
D2
D1
D0
C0
C1
C2
C3
D0
D1
D2
D3

Fig. 13

3.2.3. Addressing Scheme.

For the next few sections, let the switches on the network tree be numbered according to the following algorithm (See Fig. 14):

Let *A* be a sting representing a binary number.

(B) The root switch is assigned the empty address (*A*=""), i.e., the root switch has no address.

(R) For all other switches or stations: Let **P** denote the parent of a switch or station S, and let *A* represent the address of **P**. For all children S of **P**: let the leftmost child of **P** have the address 00*A*, the left center child the address 01*A*, the right center child have the address 10*A*, and the rightmost child have the address 11*A*. (Example: If ß is a switch with *A*= "0110," then ß's left center child would have the address 01*A*, or "01***0110***").

In a section that follows, it will be shown how this address assignment simplifies packet routing and reduces switch size and processing time.

3.2.4. Packet Format.

This implementation of a CAMB Tree Network has a packet format with three fields: a *Routing Field*, an *Address Field*, and a *Data Field.*

Packet Header----------------------------|

Routing Field	Address Field	Data Field

CAMB packet format

The Data Field contains the packet information. This field may also include some higher level protocol functions such as error checking.

The Address Field contains a packet's destination address. Receiving stations use this field to determine if a packet is for them. The number of bits in the address field depends on the size of the network. Since the address field is only valid for

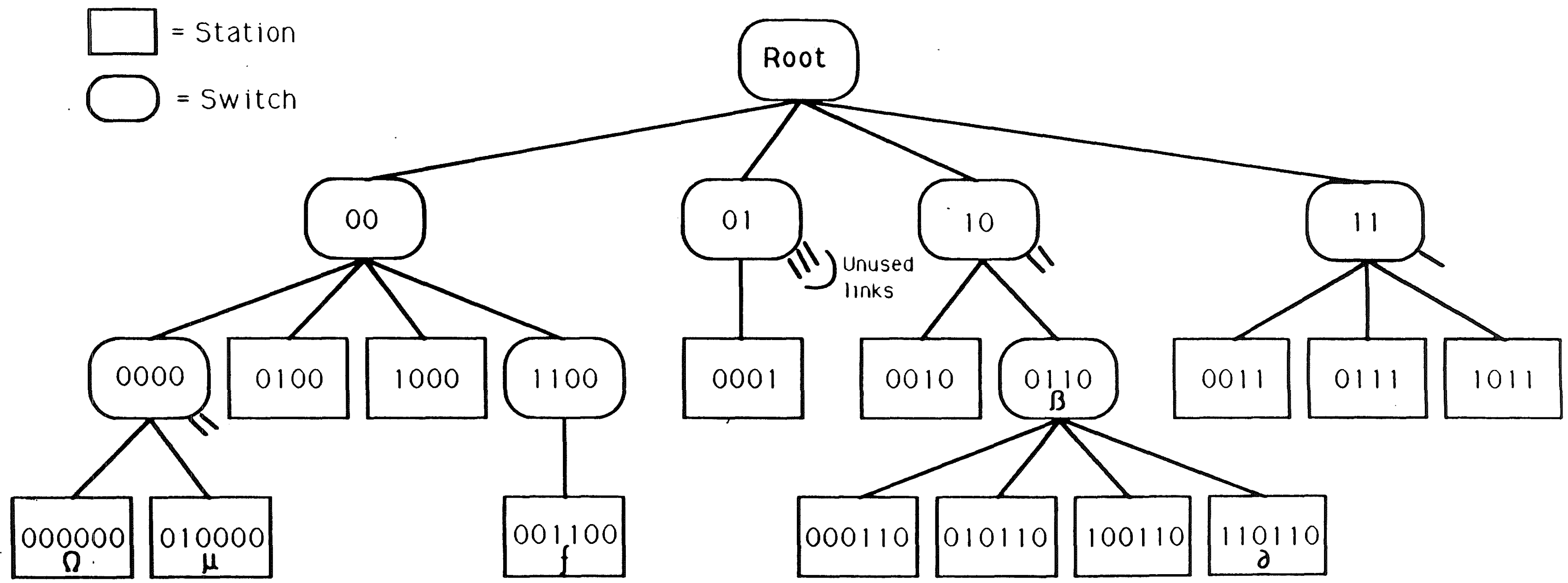


Fig. 14: Addressing Scheme.

destinations within the tree network, this field might be empty for a packet which is going to leave the tree network through the network's root switch.

The Routing Field which starts each packet is a very special entity. It is responsible for correctly switching a packet through the tree network. At each switch in the network, a single binary decision is made: 0)switch the incoming packet to the parent uplink, or 1)switch the packet to the Downlink Selector for broadcast. In the TTL implementation presented here, the first bit in the data field of a newly selected packet makes this decision. If this bit is a logical zero, all other bits in the packet are switched to the parent uplink. Otherwise, the rest of the packet is routed to the Downlink Selector. Note that in either case, the first bit of the packet is *absorbed* by the switch.

A packet's Routing Field consists of a number of *Routing Bits* equal to the number of levels the packet must ascend to reach the least common ancestor of the source and destination station(s). And of these Routing Bits, only the last bit is a one.

For example, if a packet must ascend four levels of switches to get to the correct least common ancestor, the packet would tell the first four switches to send it up the tree, and the fifth switch to broadcast it back down the tree. In other words, the Routing Field for this packet would contain "00001."

The Routing Field generation can be accomplished by comparing the source and destination switches' addresses. It is at this point that the previous assignment of switch addresses comes in handy.

Look again at figure 14. If a packet is sent from station μ to station ∂, the packet must pass through the root switch. This is because the root is the least common ancestor of μ and ∂. Notice that the addresses of these stations do not share a common postfix. However, the least common ancestor of μ and f is the station labeled "00." Note that here, the stations' addresses share the last pair of bits, "00." Consider further a transmission from Ω to μ. Now, the common ancestor is "0000," and the addresses of Ω and μ share the two rightmost bit pairs ("xx 00 00").

In fact, due to a property of the addressing scheme, for any two station addresses, the number of identical rightmost bit pairs will be equal to the level of the least common ancestor of the two stations. (Here, we define the level of a switch or station in the network tree such that: (B) the root switch is at level 0. (R) Any other switch in the tree has a level equal to that station's parent's level plus 1.)

For example, if station S1 has an address of 010101110000, and station S2 has an address of 001100110000, we can see that the three rightmost bit pairs are the same for both addresses:

S1: 01 01 01 ***11 00 00***
S2: 00 11 00 ***11 00 00***

From this, it can be concluded that the least common ancestor of S1 and S2 is on the third level of the network tree. Further, since S1 has an address consisting of six pairs of bits, it can be deduced that S1 is on level 6 of the tree. This means that a packet sent from S1 to S2 would need to ascend 6-3 = 3 levels to reach a common ancestor. Therefore, the Routing Field would contain three bits: "001."

The calculation of the routing bits may seem costly at first glance, however, initial figures indicate these bits can be generated quickly by a small amount of hardware (4-5 chips) in a modest number of clock cycles (averaging to the height of the tree network).

3.2.5. Routing Field Advantages.

Although their generation requires a small pre-processing step for each packet , the use of routing bits has several distinct advantages:

1) Each packet is quickly processed at every switch node. The switch does not have to wait to see an entire address before the decision to broadcast or not is made. This advantage becomes more prominent as the number of stations (and, thus, the length of the addresses) increases.

2) The number of routing bits in a packet depends on the number of switches the packet needs to pass through. If a packet does not have to go all the way to the network's root node, the number of routing bits will be equal to the number of levels up the tree the packet travels. Again, this makes for a very fast packet travel time through the network. So by moving some of the processing work to the stations, packets busy the network for a shorter time.

3) The use of routing bits simplifies switch design tremendously. Less components are needed to process one bit than to process an entire address. Less components mean a lower switch node construction cost and higher speeds.

4) Flexibility. All configuration is done at the network stations. The switches themselves will accommodate any tree topology desired.

5) Expandability. If it should become necessary to increase the number of levels in the network, the switch design need not be changed. Since the network itself does not deal directly with the destination address, the only modification that needs to be made is at the transmitting station. If the routing bit generation algorithm is software based, this means a modification in the generation program is all that is needed to expand the network.

6) Since the switch nodes do not deal directly with the destination address, it is possible with a slight change in packet format (adding one bit) to create a virtual addressing system with multiple-destination transmission capability. It is easy under this design to introduce address aliasing which will be recognized by more than one receiving station. This feature would be useful to users interested in conferencing over the network.

and 7) Once the sending station knows the number of levels in the tree network, the station can easily send a packet in a foreign format out of the tree network.

3.2.6. The Tree Switch Signal Protocol

In this implementation of a CAMB Tree Switch, every uplink or downlink has a *control path* and a *data path*. This makes four different signals possible in all. Below is a table summarizing what these signals mean to the switch. Notice that the signals on the downlink line do not all have the same meanings as the signals on the uplink line.

X = don't care ≠ = positive edge clock transition.

Signals on Uplinks	Data Line	Control Line	
Idle Line/End of Xmission	X	0	
Start of Xmission	X	≠	
Valid Xmission	X	1	
"Dummy Packet" transmission.	0	1	for *entire*

Signals on Downlinks	Data Line	Control Line
Idle Line/End of Xmission	0	0
Start of Xmission/Interrupt	1	0
Valid Xmission	X	1

On the uplink line, a new packet is detected by a positive edge transition of the control line. The control line remains high for the entire packet transmission. Then, when the control line goes low the switch resets and is ready to receive another packet.

On the downlink line, notice that the signal (Data=1, Control=0) is no longer considered an idle state, but signals either the start of a packet transmission, or an interrupt. This control-data bit pair is actually the last routing bit. Only now, it represents a start bit on the downlink lines. A receiving station knows a complete packet consists of a start bit (Data=1, Control=0), data, and then the idle line state. A receiving station knows a packet was interrupted if another start bit appears before the idle state (Data=0, Control=0) is reached.

3.2.7. The Circuit.

In figure 13, the entire tree switch node is drawn. This paper will conclude now with a brief rundown of the different parts of the implementation.

On the lower half of the page is the Uplink Selector:

- The 74273 at the bottom of the page is an eight-bit latch, and it receives the input from the switch uplinks. The latch functions as the Synchronizer in the Uplink Selector. The resistor array just before the latch pulls down to zero (the idle state) any unconnected inputs.

- The two 7474 chips directly above the 74273 function as the Start Detector. They are duel positive-edge flip-flops which detect the rising edge of each control line at the start of a packet transmission.

- The 74148, the OR gates, and the 74175 function as the Priority Resolver. The 74175 is a quad positive-edge flip-flop with common clock. The 74175 and the OR gates act as a filter, shifting which line has priority at any given time. The 74148 is a priority encoder which outputs the number of the selected uplink via Mux1, Mux0.

- The 74153 in the middle left of figure 7 is the Multiplexer which passes only the selected packet.

The upper half of the page contains the Address Recognizer and the Downlink Selector:

- The 7474 in the top left hand corner of figure 7 stores the first bit of the routing field. It acts as the Routing Bit Decoder.

- The Packer Router and Resynchronizer are combined in the 74399, which is a latched multiplexer.

- Below the 7474 is a group of logic gates. These gates implement the function that resets the switch. In other words, these gates form the End Recognizer.

- Lastly, The 74153 in the upper right corner of figure 7 acts as the Downlink Selector.

4. Further Research.

This paper puts forth two TTL circuits which implement a CAMB Tree network. The switch described in the implementation #1 has already been built and simulated. The circuit performed well under simulation and the prototype is working according to the specifications.

The second circuit has performed well under simulation, but there is one change in the original protocol of the CAMB circuit which might be conducive to a more efficient operation. When a packet is being broadcast from the parent downlink of a switch, the Uplink Selector might still try to send a packet to the Address Recognizer which is destined to be broadcast. Yet, the broadcast line is busy because the parent downlink line is transmitting a packet. It would be a simple modification to have Uplink Selector choose from the set of upward traveling packets (packets which are not destined to be broadcast at this switch) rather than from all packets available.

A second piece of research in progress is to find the most efficient routing bit generation algorithm. The current version is not excessively complex, but there may be room for improvement. An actual hardware circuit for generating the routing bits is currently under development.

Finally, the TTL implementation given here uses two lines, a control line and a data line, for every uplink and downlink. A one-line version of this implementation is being constructed using the logic simulation program which may have some cost advantages over the two line version.