Report Number: WUCS-89-52

1989-10-01

Design of a VLSI Broadcast Translation Circuit

Gaurav Garg

This paper describes the design of a Broadcast Translation Circuit chip. The Broadcast Translation Circuit (BTC) provides unique addresses for each of the copies of a broadcast packet replicated by the copy network in a Broadcast Packet Switch [Tu85], [Tu86]. During a packet cycle or epoch, a packet is then passed on to the routing or distribution network. This chip has been fabricated in 2.0 µm CMOS technology.

DESIGN OF A VLSI BROADCAST TRANSLATION CIRCUIT

Gaurav Garg

WUCS-89-52

October 1989

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

ABSTRACT

This paper describes the design of a Broadcast Translation Circuit chip. The Broadcast Translation Circuit (BTC) provides unique addresses for each of the copies of a broadcast packet replicated by the copy network in a Broadcast Packet Switch [Tu85], [Tu86]. During a packet cycle or epoch, a packet is accepted from the copy network suitable translation is performed on the packet header, and the packet is then passed on to the routing or distribution network. This chip has been fabricated in 2.0 μm CMOS technology.

This research was sponsored in part by funding from NSF Grant DCR-8600947, Bellcore, BNR and Italtel.

Table of Contents

List of Figures

Design of a VLSI Broadcast Translation Circuit

1. Introduction

This paper describes the design of a VLSI Broadcast Translation Circuit (*BTC*) for use within a broadcast packet switching network. This network is described in [Tu85], and [Tu86].

1.1. Packet Switch

The overall structure of the prototype packet switch is shown in Figure 1. The switch terminates 15 fiber optic links. A production version would be much larger. The packet switch is divided into five major components. These are the Connection Processor *CP*, the Packet Processors *PP*, the Copy Network, the Broadcast Translators and the Routing Network. The Copy Network, Broadcast Translators, and Routing Network are collectively known as the Switch Fabric (*SF*), and both networks are constructed from banyan-interconnected Switch Elements. The *CP* is responsible for establishing connections. It exchanges control information with Connection Processors in neighboring Switch Modules and controls the actions of the Packet Processors and the Switch Fabric. The Packet Processors serve as buffers between the transmission links and also perform the address translation required for routing. The Copy Network's function is to make copies of broadcast packets as they pass through it. Point-to-point packets pass through the Copy Network without change. When copies of a broadcast packet emerge from the Copy Network, their final destinations are yet to be determined. The Broadcast Translators assign a new routing field to each copy of a broadcast packet in such a way that a copy is received by every *PP* that is supposed to receive one. The Routing Network then routes the packets to the appropriate output links based on information encoded in the routing fields of the packets.

1.2. BTC functionality

The copy network *CN* makes copies of broadcast packets in such a way that if any particular output port of the *CN* receives a copy then the identity of that copy is known. The final destination of the copy is yet to be determined. The *BTC* provides a new routing field to determine the output *PP* for the given copy depending on the broadcast channel number and the copy number. The copy number is determined by the output port to which the *BTC* is connected. The *BTC* performs the address translation by a simple table lookup. Each *BTC* contains a Broadcast Translation Table (*BTT*) for this purpose. The *BTT* is indexed by the Broadcast Channel Number (*BCN*) of the incoming packet. Thus each *BTC* provides a unique output link number for a given *BCN* corresponding to a particular copy of the broadcast packet.

The *CP* controls the information stored in the Broadcast Translation Tables. It writes 4 bytes to each *BTT* entry. These are used to replace to provide a new 4 byte header for broadcast packets. These four bytes contain a new routing field and an output link number which is based on what the copy number would be if a copy was received at that *BTC*, and it is used to route the packet through the Routing Network (*RN*). The *CP*

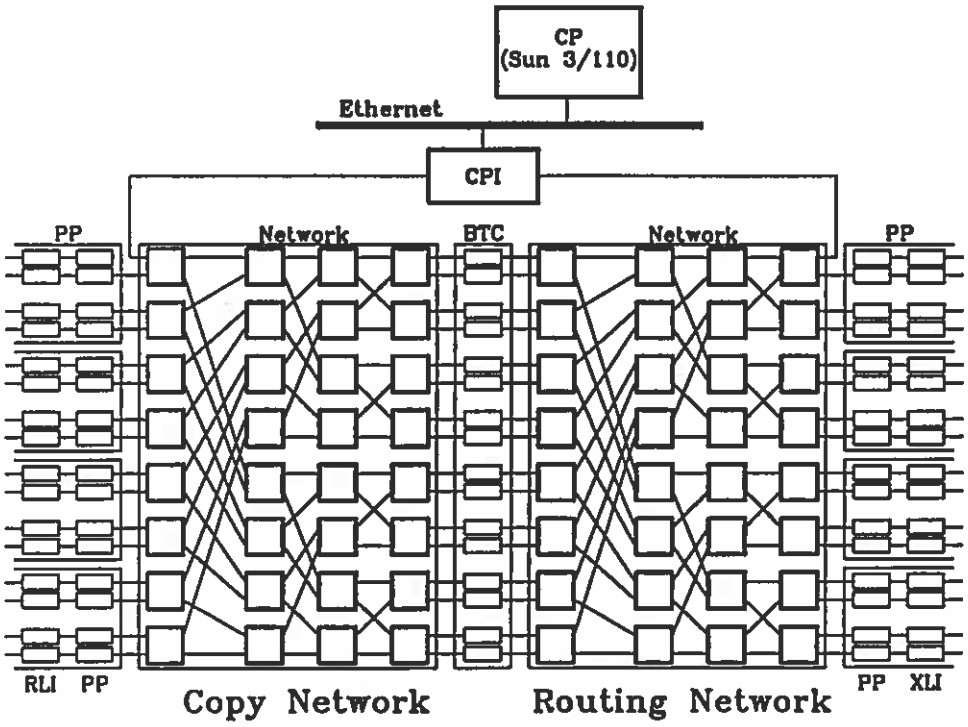

Figure 1. Switch Module

may examine an entry in the *BTT* without making any change.

1.3. Packet Formats

This section describes the packet format relevant to the *BTC*. A more complete description may be found in [Ba88]. There are two primary packet formats: external and internal. Packets are carried in external format on the fiber optic links connecting switches, and in internal format within each switch. The *PP* translates between these two formats. Fig. 2 depicts the internal packet format. The *BTC* does not interact with external packets.

Each internal packet is organized as a sequence of nine bit wide words, including an odd parity bit. Each packet contains exactly 80 words, of which 72 make up the information field. The control fields contained in the internal packet format are as follows :

- Routing Control - *RC* - pt-to-pt, broadcast, specific-path
- Operation - *OP* - variety of control functions
- Fanout - *FAN* - number of output ports with copy of packet
- Link Number - *LN* - number of outgoing link for pt-to-pt packets

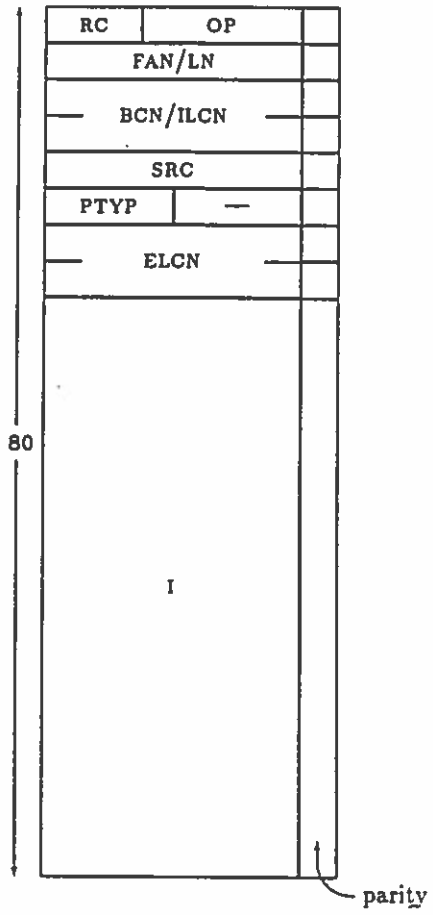

Figure 2. Internal Packet Format

- Broadcast Channel Number - *BCN* - id for multipoint channel
- Internal Logical Channel Number - *ILCN* - id for pt-to-pt channel
- Source - *SRC* - number of most recent packet processor

The *SRC* field passes through the *BTC* unchanged. The following describes how the other header fields described above interact with the *BTC*.

1.3.1. Routing Control - *RC*

This 3-bit field determines how the packet is processed by the switch elements. It is used by the *BTC* to distinguish between broadcast packets and point to point packets. The four possible interpretations are listed below.

- Empty Packet Slot - 0

- Point-to-Point Data Packet - 1
- Broadcast Packet - 2
- Specific Path Packet - 3

1.3.2. Operation - *OP*

This 5-bit field specifies control operations. Table 1 lists the various codes and their interpretations. The Read and Write BTT codes are control functions that affect *BTC* operation.

Table 1. List of Operations in the *OP* field

Code	Field Name
0	VANILLA
1	Read LCXT Entry
2	Write LCXT Entry
3	Read BTT Entry
4	Write BTT Entry
5	Switch Test Packet Leg 1
6	Switch Test Packet Leg 2
7	Switch Test Packet Leg 3

The *OP* fields that interact with the *BTC* are described below.

1.3.2.1. Vanilla Packet

This implies that there are no control instructions in this packet. This code is attached to data packets.

1.3.2.2. Read BTT Entry

This directs the *BTC* to read and return a single entry from the Broadcast Translation Table *BTT*. The I[0] field specifies the address of the entry, and the output data is copied onto I[1]-I[4]. The packet is then returned to the connection processor.

1.3.2.3. Write BTT Entry

This directs the *BTC* to write information to a single entry of the *BTT*. The I[0] field specifies the address to write to, and the input data is in I[1]-I[4].

1.3.3. Fanout/Link Number - *FAN/LN*

This 1-byte field either provides the number of output ports for copies of a Broadcast Packet or the outgoing link number for point to point packets. The *BTC* writes to this field to provide an outgoing link number for the copy of a Broadcast Packet. The *BTC* sets this field to zero for *RBTT* packets, since port zero is dedicated to the connection processor which made the *RBTT* request.

1.3.4. Broadcast Channel Number/Internal Logical Channel Number -*BCN/ILCN*

This 2-byte field provides the *BCN* for broadcast packets or the *ILCN* for other packets. The *BCN* is used to index the look-up table in the *BTC*, which then replaces the first four bytes of a packet, thus providing an output port for the copy of a broadcast packet.

1.3.5. Information - *I*

This 75-byte field carries user data for data packets. If the packet is a *RBTT* or *WBTT* packet, then the first five bytes of this field are used to carry the address and data for the *BTT*. One byte is reserved for the address, and four for the data to the look-up table.

1.4. Broadcast Translation Circuit - *BTC*

The *BTC* provides unique addresses for each of the copies of a broadcast packet replicated by the copy network. A simplified block diagram of the *BTC* is shown in Figure 3.

The *INC* examines incoming packets and routes necessary information to the *BTT*. For example, if the input packet was a broadcast packet, then the *BCN* of the packet would be used to address the *BTT*. The *BTT* has 64 entries, each of which consists of 4, 9-bit words. The *delay* circuit delays packets to allow time for *BTT* lookup. If the input packet is a broadcast or a *RBTT* packet, then the *OUTC* replaces 4 bytes in the packet with output data from the *BTT*.

All of the internal operation of the switch occurs in a very synchronous manner. The clock speeds on the surrounding packet switch elements and the *BTC* are the same, and thus all of the operations of the *BTC* may be performed in a synchronous manner with no buffering necessary. Some delay circuits (mini-FIFOs) are introduced for internal synchronization, but there is no buffering of an entire packet at any stage. The switch fabric serial data speed is about 25 *MHz*, but the *BTC* is designed to operate at speeds up to 50 *MHz*. The remainder of this paper is organized as follows :

Section 2 presents the chip specification. Section 3 describes the design of various circuit components. Section 4 describes in detail the design of the *BTT*. Section 5 describes the functional and timing simulation performed on the circuit representation. The main body of the text is followed by a set of appendices.

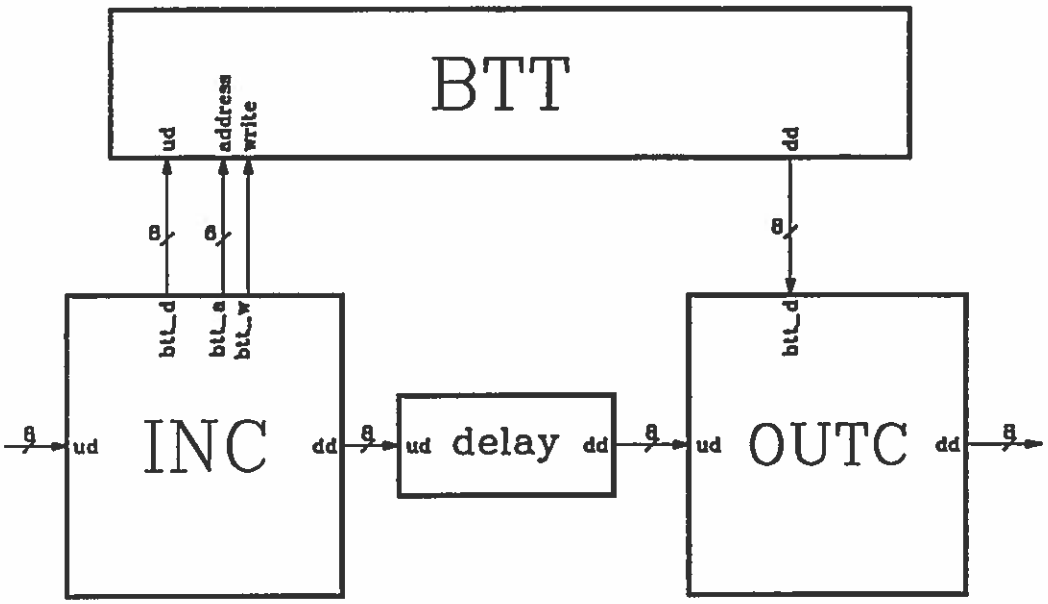

Figure 3. Simplified Block Diagram - BTC

2. Chip Specification

2.1. Operation

The *BTC* operates on the basis of a packet cycle which starts on a *pt* signal. The chip is clocked by a two phase non-overlapping clock. The desired clock rate is 50 *MHz* giving a clock period of 20 *ns*. When $\phi 1$ is high, signals on the upstream lines are valid. The *pt* line goes high when $\phi 2$ is high and stays high for one clock cycle. Successive upgoing transitions of *pt* are at least 86 clock cycles apart. The system maintains odd parity. All data packets experience a delay of 32 clock cycles while passing through the *BTC*. Point-to-point packets pass through unchanged, whereas broadcast packets have the first four bytes replaced. Control packets are dealt with differently and will be described later in this document.

2.2. Interface

The specification for the interface to the *BTC* is described here. The external interface signals for the *BTC* are shown in Figure 4 and are described briefly below.

2.2.1. Upstream Data - *ud*0 .. *ud*8

This is the incoming data from the upstream packet switch elements. It consists of 8 data bits *ud*0 .. *ud*7 and 1 parity bit *ud*8. The data arrives when $\phi 2$ is high and is valid when $\phi 1$ is high. Upstream data arrives 4 clock cycles after the *pt* signal is asserted.

2.2.2. Downstream Data - *dd*0 .. *dd*8

This is the outgoing data to the downstream packet switch elements. It also consists of 8 data bits and 1 parity bit in the same format as that for upstream data. Downstream data appears 36 clock cycles after the *pt* signal is asserted.

2.2.3. Packet Time - *pt*

This signal indicates the beginning of a packet cycle. It goes high when $\phi 2$ is high, is valid for a high $\phi 1$, and remains high for a total of one clock cycle. As mentioned before, successive *pt* signals appear at least 86 clock cycles apart.

2.2.4. Hardware Reset - *hrst*

This signal resets the chip. It clears all internal counters that synchronize internal operation, however it does not clear the contents of the look-up table.

2.2.5. Soft Reset - *srst*

This signal clears the *perr* signal after a delay of one clock cycle. It does not affect any of the internal operation of the circuit.

2.2.6. Parity Error - *perr*

The circuit computes parity and compares it to the corresponding parity bit in two places. The first parity checker is placed where upstream data is received, and the second is at the output of the look-up table. If either path has a discrepancy between the computed parity and the associated parity bit, then the *perr* bit is asserted.

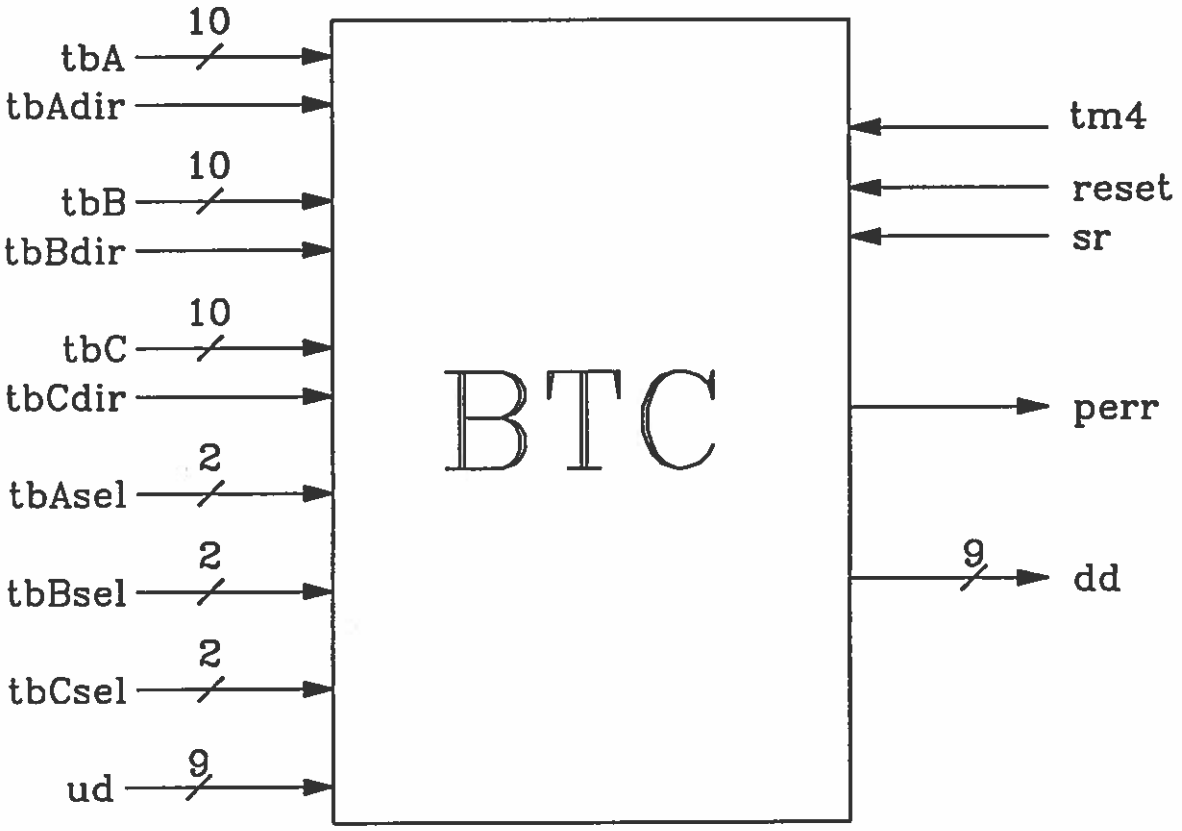

Figure 4. Broadcast Translation Circuit Interface

2.2.7. Test Bus Leads - *tbA* 0 .. *tbA* 9, *tbB* 0 .. *tbB* 9, *tbC* 0 .. *tbC* 9

The *BTC* was designed in a manner that would facilitate testing of the internal components and buses. There are three test buses which have pins on the chip packaging which allow the user to monitor internal data buses or to test the integrity of internal circuit components by driving data (for which there is known output) onto the internal bus preceding the component. Each bus has a 10-bit wide path, and is connected to various test access points which control the data provided to the buses. A *Tap* is placed on an internal data bus, and has an upstream side at which data is received, a downstream side where data continues on the internal bus, and a test bus side where it is connected to the test bus. Data on the bus must pass through the *Tap* and is subject to change depending on the configuration of the *Tap*. The buses may be labeled bus A, bus B, and bus C for reference. A later chapter will explain how the test access points work, and the relationships between them and various internal buses.

2.2.8. Test Bus Selection Leads - *tbAs* 0 *tbAs* 1, *tbBs* 0 *tbBs* 1,*tbCs* 0 *tbCs* 1

Each test bus may be connected to up to four test access points. A 2 bit address must be provided for each test bus to pick one test access point. The address provided will result in enabling the selected *Tap*, which implies that it will interface with the test bus. The remaining test access points do not interact with the test bus. They are transparent to internal data.

2.2.9. Test Bus Direction Leads - *tbAd*, *tbBd*, *tbCd*

Each *Tap* has two modes. One mode monitors, and the second drives the internal bus to which the *Tap* is connected. Data is not affected in the "watch" mode. In the "Drive" mode the contents of the test bus is put on the internal bus. A detailed description on how this is used will appear in a later chapter.

2.3. Global Timing

Figure 5 shows the relationships between the clocks and data during a clock cycle for a clock rate of 42 *MHz*. Notice that $\phi 1$ is high for 40% of a clock cycle whereas $\phi 2$ is high for 20% of the clock cycle.

Figure 6 shows the flow of upstream and downstream data with respect to the *pt* signal. The test bus select and direction leads maintain a constant value throughout any run of the circuit. Changes in data on the test buses occur in synchronization with upstream and downstream data.

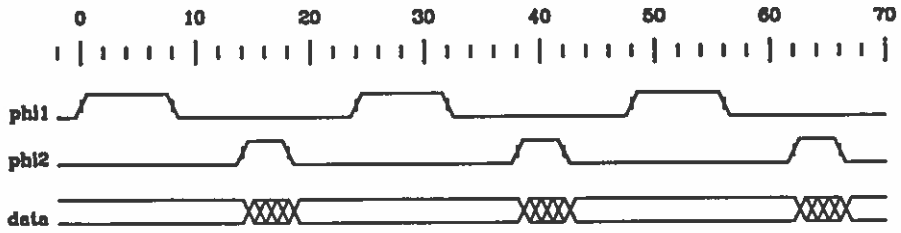

Figure 5. Clock and Data Relationship within a Clock Cycle

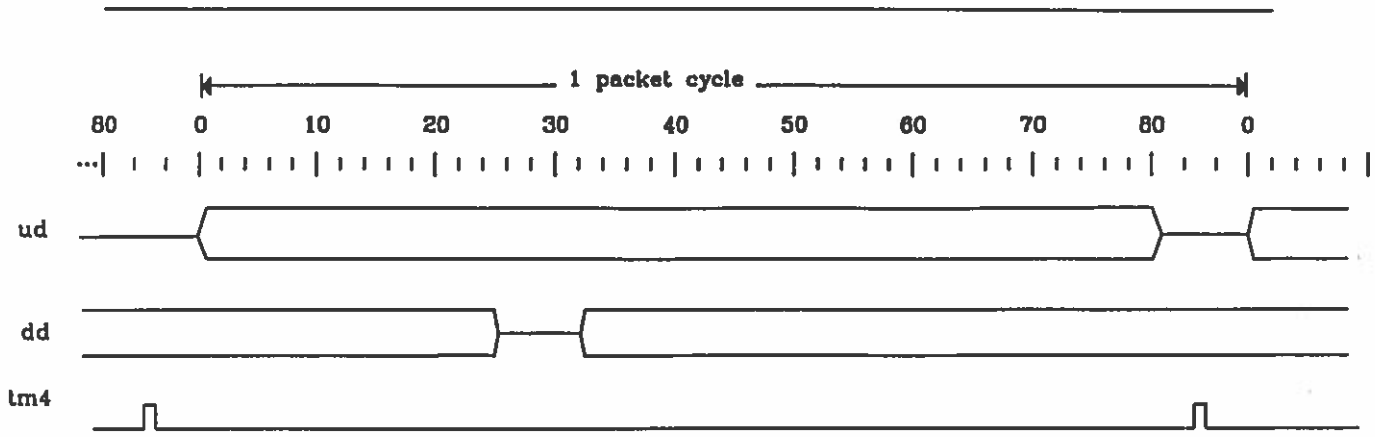

Figure 6. Upstream and Downstream Data Flow

3. Component Design

3.1. BTC - Overall System Mechanism

Figure 7 shows a detailed block diagram of *BTC*.

3.1.1. Data Packets

When a broadcast packet arrives at *INC*, the *BCN* is used to address the *BTT*. The *BTT* uses this address to look-up an output port (it outputs a 4 byte header). The packet is delayed in the *delay* circuitry while the look-up is being performed. The *OUTC* replaces the first 4 bytes of the packet with the 4 bytes received from the *BTT*. Other types of data packets pass through the *BTC* unchanged with a delay which is the sum of the delays through the *INC*, *delay*, and *OUTC* blocks.

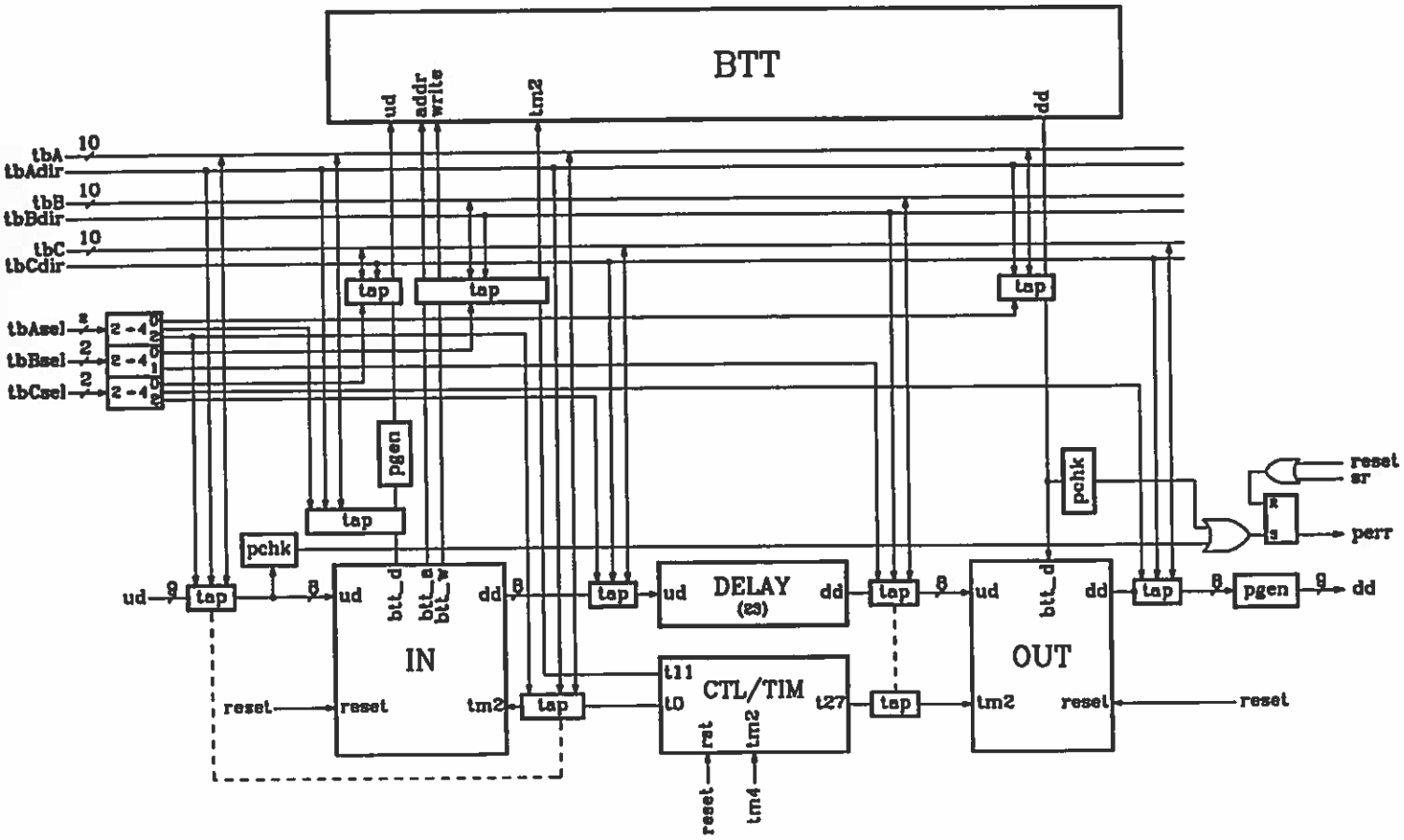

Figure 7. BTC - Detailed Block Diagram

3.1.2. Control Packets

The *BTC* passes all controls packets except *WBTT* and *RBTT* unchanged. The *WBTT* and *RBTT* packets use the first byte of the information field *I* to store the address sent to the *BTT*, and the next four bytes to store the data sent to or received from the *BTT*. If the packet is a *WBTT* packet the *INC* asserts a *write* input lead to the *BTT*, and also sends it a 4-byte data field with a 6 bit address field indicating the *BTT* address (0 to 63) to which the data should be written. The *RC* field gets set to 0 indicating a null packet, but the rest of the packet passes through the *BTC* unchanged.

If the packet is a *RBTT* packet, the address stored in the *I* field is sent to the *BTT* which outputs the requested 4-btye contents of the given address. The *OUTC* replaces the four bytes of the *I* field reserved for *BTT* data in a *RBTT* packet with the 4-bytes received from the *BTT*. The *OUTC* also sets the *RC* field to point-to-point and the output link number *fan_ln* to 0. This sends the packet to output port 0 which is connected to the *CP*.

3.2. Integrated Circuit Design

The circuits for the *BTC* were designed using a set of CAD tools from the University of California at Berkeley. The circuit layout tool was MAGIC, a graphical layout editor supporting a hierarchical design style. The technology based design rules are checked by the editor, and are based on a Mead-Conway design style with $\lambda = 1.0\,\mu m$. This results in a minimum line width of $2\,\mu m$. Chip fabrication is a bulk n-well, 2 layer metal, single poly CMOS process, using facilities provided by MOSIS.

3.3. SSPC - Circuit Generator

The detailed block diagram in Figure 7 shows all the functional blocks in the circuit. The *INC*, *OUTC*, *delay*, and *ctltim* blocks were generated via a tool called *SSPC* developed by George Robbert [Ro88]. This tool allows the user to specify a high level description of a certain class of circuits and it generates the mask level layout for the required circuit. The *INC* and *OUTC* blocks were created using the *SSPC*. The delay line was generated using a tool called *peg_delay* which is part of the *SSPC*, generating a delay line of appropriate width and delay in number of clock cycles. The control and timing circuit was generated via a tool called *peg_ctlt*, also part of the *SSPC*. The *BTT* has an internal control and timing circuit that was also generated using *peg_ctlt*.

3.4. Global Control and Timing - CTLTIM

The *INC*, *OUTC*, and *BTT* all work on a packet cycle, but start their internal packet cycle at different times with respect to the global packet cycle. The *ctltim* is a global counter that counts through a packet cycle. It keeps track of the global timing and synchronizes the operation of the entire *BTC*. It has three output signals, each a local *pt* signal to one of the three circuits mentioned here. The circuit asserts the appropriate *pt* signal when its internal counter reaches the preset value at which the circuits it synchronizes start their local packet cycles. Logical time zero in a control and timing circuit is two clock cycles after the local *pt* signal is asserted for one clock cycle. The input to the *SSP* generator for the global *ctltim* is provided in Figure 8.

The value of the counter in the global control and timing circuit will be used as the logical clock cycle number for *BTC*. The local *pt* signals to *INC*, *OUTC*, and *BTT* are

```
/* Input Description for global control and timing circuit - BTC
 */
        condinps  C0
        muxouts  inc_tm2 outc_tm2 btt_tm2
        cntsize 7
        cntmax 90
        cntena  - 86 <MAX>
        sigset   - 0 inc_tm2
        sigrst   - 1 inc_tm2
        sigset   - 27 outc_tm2
        sigrst   - 28 outc_tm2
        sigset   - 11 btt_tm2
        sigrst   - 12 btt_tm2
```

Figure 8. Global Control & Timing Circuit Description

*inc_tm*2, *outc_tm*2, and *btt_tm*2. The circuit counts up to 86 clock cycles and the count stays at 86 until the next *pt* signal which sets the count to zero. The *pt* time signals sent to the major blocks in the circuit are asserted only for one clock cycle as the input to the generator shows. The circuit is of size 452 λ X 626 λ, and has 416 transistors.

3.5. Input Circuit - INC

The *INC* is an *SSP*. The function of the circuit is to receive a packet and output portions of the packet to various output ports when the value of its logical clock reaches associated predetermined values. It receives data from the upstream ports on the *BTC*, interacts with the *BTT*, and sends the data to the delay line. The input to the *SSP* generator for the *INC* is provided in Figure 9.

The upstream data *ud* to the circuit is an 8-bit wide port and valid data arrives at logical time 0 which is two ticks after the local *pt* signal is asserted. Downstream data *dd* that goes to the delay line, is also an 8-bit wide port and it appears at logical time 4. The write signal to the *BTT* is valid for 16 clock cycles starting at logical time 7. The address to the *BTT* is an 8-bit wide port and it is valid during clock cycle 15. The 4-byte data to the *BTT* is sent along an 8-bit port and it is valid during clock cycles 14, 15, 16, and 17.

The default values for the signals to the *BTT* appear in the *SSP* description. In the case that the packet is a broadcast packet the first condition is satisfied, and the *BCN* is diverted to the address port going to the *BTT*, and the *RC* field is set to pending to alert the *OUTC* that it is expected to perform a header replacement. If the packet is a *WBTT* packet, then the *RC* field is set to zero and the *BTT* write signal is asserted. The circuit is of size 1588 λ X 2991 λ and has 4376 transistors.

```
/* Synchronous streams processor definition for the input circuit for
/* the BTC chip.
*/

#include "pktfmt.sih"

inc@86(port[8] int_pkt      <ud@0;      /* data from CN */
       port[8] int_pkt      >dd@4;      /* data to delay */

       port[1] bit[16]      >btt_w@7;   /* BTT write signal */
       port[8] bit[8]       >btt_a@15;  /* address to BTT */
       port[8] bit[4][8]    >btt_d@14)  /* data to BTT */
{
    dd = ud;
      btt_w = 0;
      btt_a = ud.extp.info[0];
      btt_d = ud.extp.info:(8..39);

      if ud.rc == MPNT & ud.op == VANILLA ->
        btt_a = ud.BCN_ilcn:(0..7);
        dd.rc = PENDING;
       | ud.rc != IDLE & ud.op == WBTT ->
        dd.rc = 0;
        btt_w = 0xffff;
      fi;
}
```

Figure 9. SSP Description for INC

3.6. Output Circuit - OUTC

The *OUTC* integrates data from the delay line and the *BTT*. The *SSP* description for *OUTC* is provided in Figure 10.

If the *RC* field of the packet arriving on the upstream data side is pending, that implies that the packet expects a new routing field from the *BTT* and thus the first four bytes of the packet are replaced by data from the translation table.

If the packet is a *RBTT* packet, then the *RC* field is set to point-to-point, and the packet is directed to port 0 on the switch module output which is connected to the *CP*. The circuit is of size 1404 λ X 2807 λ and has 3942 transistors.

```
/* Synchronous streams processor definition for the output circuit for
/* the BTC chip.
*/

#include "pktfmt.sih"

outc@86(port[8] int_pkt          <ud@0;       /* data from delay */
        port[8] int_pkt          >dd@4;       /* data to DN */
        port[8] bit[4][8]        <btt_d@0)    /* data from BTT */
{
    dd = ud;
      if ud.rc == PENDING ->
        dd:(0..31) = btt_d;
     | ud.rc != IDLE & ud.op == RBTT ->
        dd.rc =.ppNT;
        dd.fan_In = 0;
        dd.extp.info:(8..39) = btt_d;
    fi;
}
```

Figure 10. SSP Description for OUTC

3.7. Broadcast Translation Circuit - BTT

Chapter 4 describes the design of the *BTT* in detail and thus it is not discussed here.

3.8. Delay Line - delay

The 8-bit wide delay line has a delay of 23 clock cycles and was generated by *SSPC*. The circuit is of size 1022 λ X 383 λ and has 1504 transistors.

3.9. Parity Checkers and Generators - pchk & pgen

The *BTC* processes an 8-bit wide data path. A ninth bit is the parity bit. The circuit checks for and generates odd parity. Parity is checked on the upstream data line of the *BTC*, and the output data line from the *BTT*. Parity is generated at the input to the *BTT*, and before data is sent out on the downstream side of *BTC*. This allows a continuous examination of the integrity of the memory in the *BTT*. It is natural to check the integrity of the input data to the circuit, and to send out data with correct parity.

A parity checker *pchk* has a 9-bit input port and an 8-bit output port. The ninth bit is the parity bit. There is an output *perr* bit which is asserted if the incoming 9-bit data did not have odd parity. The delay through the *pchk* circuit is of the order of 2 to 3 nanoseconds.

A parity generator *pgen* has an 8-bit input port and a 9-bit output port, with the ninth bit being the parity bit. The *pgen* inserts a delay of one clock cycle during its operation and requires external clocks to keep it synchronized with the rest of the system. Both *pgen* and *pchk* are simple circuits made up of cascaded *xor* gates.

3.10. Test Access System

The test access system consists of a three 2-to-4 decoders, three 10-bit wide test buses, and eight test access points (*Tap*). The three test buses A, B, and C are connected to pads on the chip packaging. The 2-to-4 decoders decide which of the test access points currently interface with the test buses. This depends on select bits *tbAs*0, *tbAs*1, *tbBs*0, *tbBs*1, *tbCs*0, and *tbCs*1 for test buses A, B, and C respectively. The test access points are used to either watch (watch mode) an internal data bus or test the integrity of an internal component (drive mode) by driving external data on to the internal bus. There is a *dir* bit associated with each bus denoted by *tbAd*, *tbBd*, and *tbCd* respectively. These bits determine the mode of the test access points connected to each bus.

3.10.1. Test Access Point - *Tap*

Figure 11 shows an interface diagram for a *Tap*.

The *dir* and *en* lines control the operation of the *Tap*. The 2-to-4 decoders described above decode the select bits and enable the selected test access points by asserting their *en* signals. All data through a *Tap* experiences a delay of 2 to 3

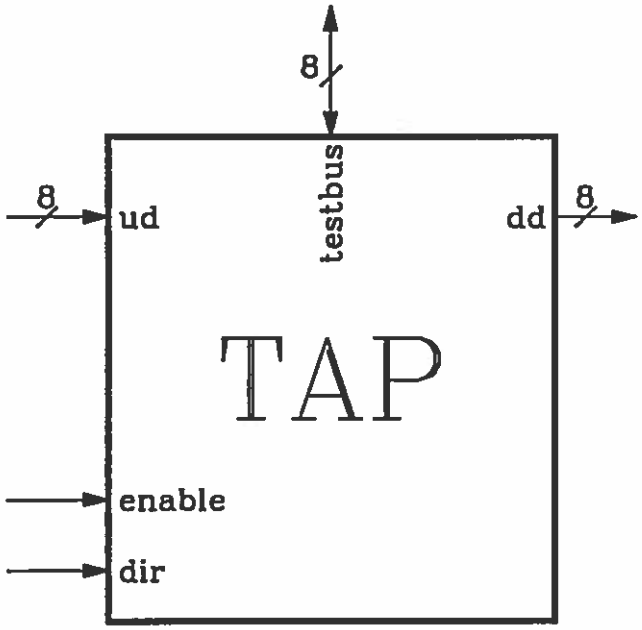

Figure 11. Interface Diagram for Tap

nanoseconds. Figure 12 shows how the *en* and *dir* signals control data flow in a *Tap*.

Figure 13 shows a block diagram of a *Tap*. Data arrives on the upstream side and in the default mode (not enabled) passes through the transmission gate. The output driver is used as an amplifier for the downstream side, and the connection on the test bus side is disabled.

If the *Tap* is enabled and in the watch mode, then not only is the transmission gate open, but also the driver placing data on the test bus lines. There is no possibility of feedback because the driver pointing the other way is disabled. Similarly if the *Tap* is in the drive mode, then the transmission gate is turned off so as not to affect data on the upstream side, and the driver that puts that data from the test buses onto the internal nodes of the *Tap* is turned on. The driver going the other way is turned off, and the data from the test buses is driven by the downstream drivers onto the downstream side of the *Tap*.

1. en = 1, dir = 0 → Transmission mode, Tap in Watch Mode

   ud → dd
   ud → tbus

2. en = 1, dir = 1 → Tap in Drive Mode

   ud → don't care
   tbus → dd

3. en = 0, dir = x → Tap not Enabled, Purely Transmission Mode

   ud → dd
   tbus = don't care

Figure 12. Data Flow in a *Tap*

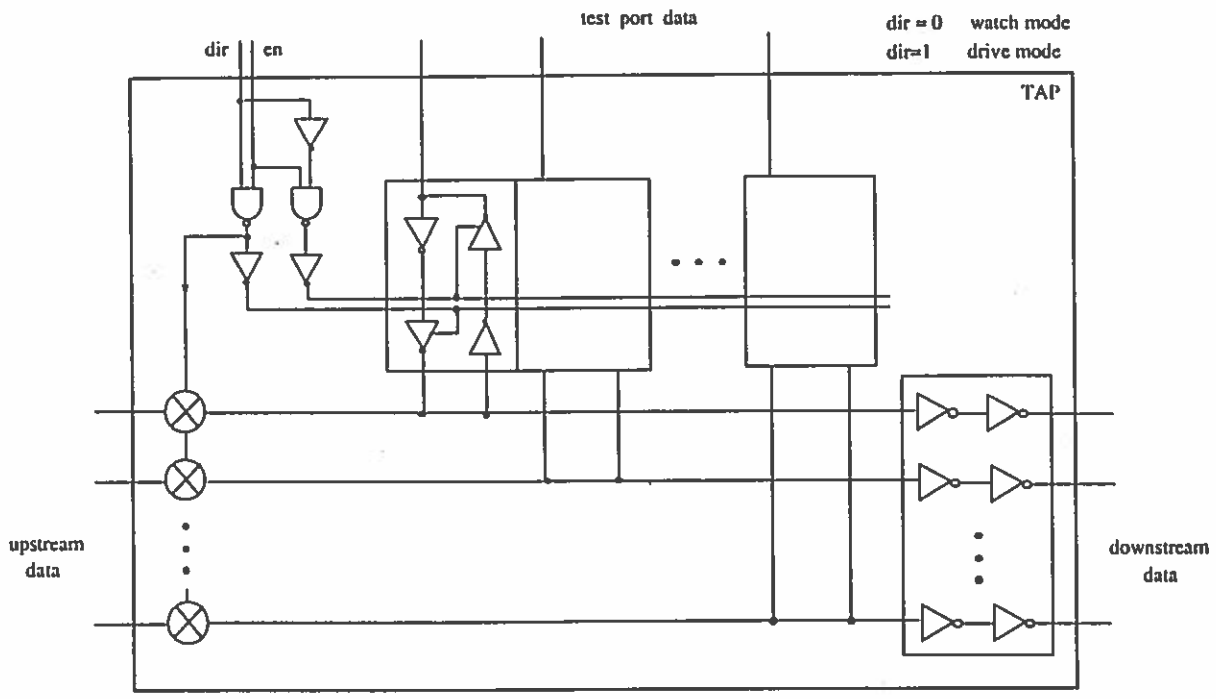

Figure 13. Block Diagram - *Tap*

3.10.2. Test Access System Organization

Table 2 shows the organization of the testing system. The entries in the table show the internal buses that the test access points are on and the port width of each such bus.

Table 2. Test Bus and *Tap* Organization

BUS	Tap # 0 : address 00	Tap # 1 : address 01	Tap # 2 : address 10
A	Outc_dd → pgen *8–bits*	Inc_dd → delay *8–bits*	pgen → btt_ud *9–bits*
B	Inc_btad→bttad *6–bits*, Inc_btrd→bttrw *bit–8*, ctltim→bttpt *bit–7*	delay→Outc_ud *8–bits*, ctltim→out_pt *bit–9*	
C	btt_dd→Out_btt (before pchk) *9–bits*	Inc_btt→Btt_ud (before pgen) *8–bits*	Ud→Inc_ud (before pchk) *9–bits*, ctltim→Incpt *bit–9*

3.11. Parity Error Bit Circuitry

The parity checkers generate two local *perr* signals. Figure 14 shows a block diagram of the circuitry used to control the *perr* signal on the *BTC*.

The *RS* FF is set by one of the two parity error signals generated by the parity checkers. It may be reset by a hardware *reset* to the entire chip, or by a *srst* whose only function is to reset the *RS* FF. The uncomplemented output of the *RS* FF is routed to a latch, the output of which is connected to the *perr* signal on the *BTC*. Thus there is a one clock cycle delay between the detection of a parity error and the assertion of the *perr* signal on the chip.

3.12. Global Timing - BTC

Figure 15 shows the sequence of events for a broadcast packet as it is processed by the *BTC*. This shows that upstream data arrives at logical time 0 with respect to the global clock, and experiences a delay of 32 clock cycles before appearing on the downstream side at logical clock 32.

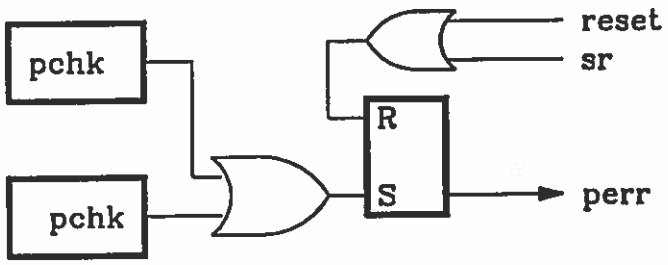

Figure 14. Block Diagram of Perr control circuitry.

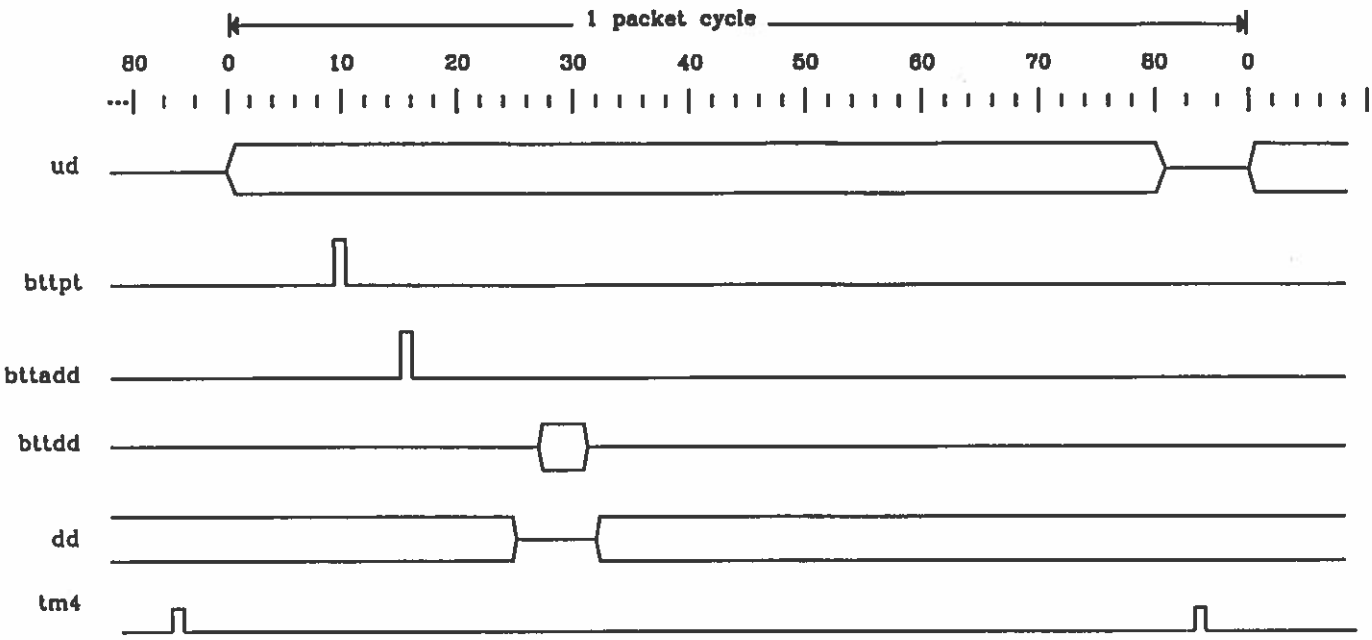

Figure 15. Global Timing of BTC for Broadcast Packet

4. Design of a Broadcast Translation Table

The *BTT* contains 64 4 word entries, where each word has 9 bits. Each of these may be addressed by 6-bit addresses. The user may write to or read from any entry in the circuit. It works on an 86 clock tick packet cycle like the rest of the *BTC*. The *BTT* is either in a read cycle or a write cycle during any one packet cycle. This is determined by the value of the *write* signal while *pt* is valid. The address is latched by the circuit at logical time 4. If the cycle is a write cycle then 4, 9-bit entries on the input are valid during logical times 4, 5, 6, and 7 respectively. If the cycle is a read cycle then 4 9-bit output words are provided during clock cycles 16, 17, 18, and 19. The reset line resets the control circuitry, but does not clear the entries in the *BTT*. The memory is made up of standard six transistor SRAM cells. The size of the circuit is 1756 λ X 4681 λ and it has 17232 transistors. It is rotated 90 degrees anticlockwise for placement in the *BTC*. Figure 16 shows a timing diagram for *BTT* operation during the read cycle. Figure 17 shows the operation during a write cycle.

4.1. Memory Cell

The memory cell is a standard 6 transistor *SRAM* cell as shown in [WeEs85]. Figure 18 shows a circuit diagram of the memory cell.

The cell stores data at points *X* and *Y*. A single *WORD* line is used to address the cell and it may be written to or read from through a pair of lines called *BIT* and *BIT'*. The user may write to a memory cell by driving the *BIT* line with the value of the data bit, and *BIT'* line with its complement. Then the *WORD* line is asserted for a given period. This causes the given bit to be saved in the cell. One may read from the cell by

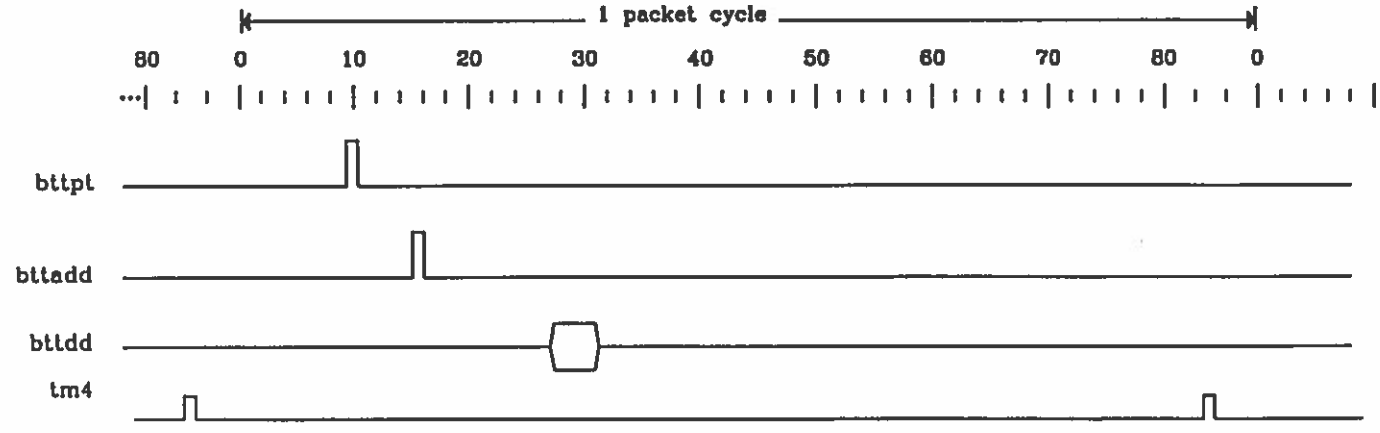

Figure 16. BTC Operation - Read Cycle

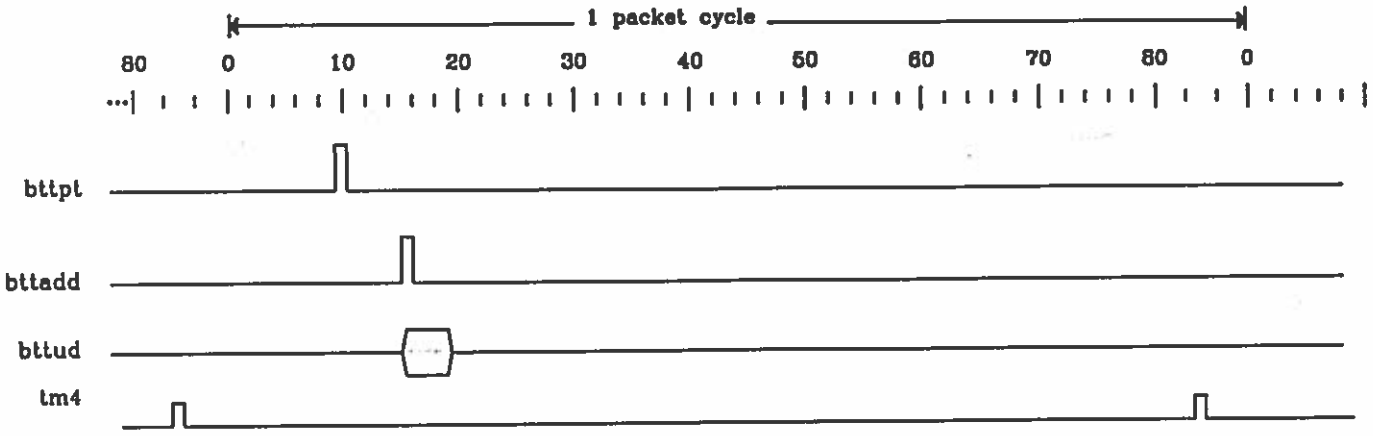

Figure 17. BTC Operation - Write Cycle

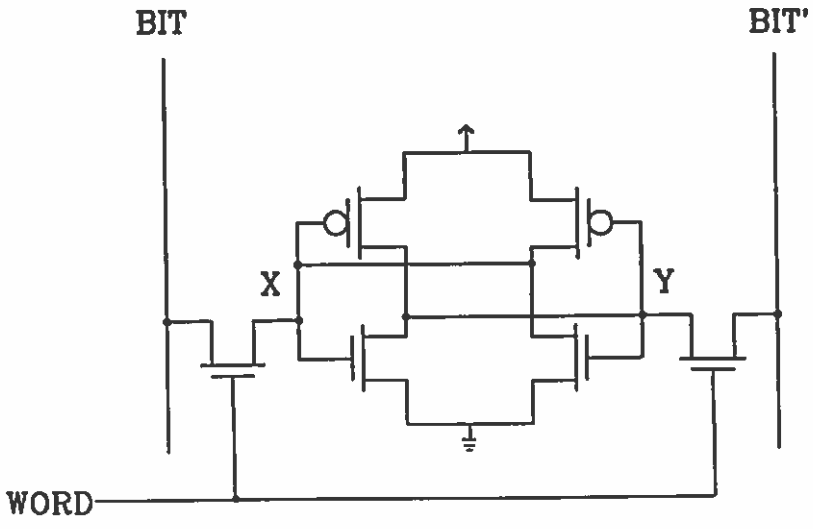

Figure 18. - 6 Transistor SRAM Cell

precharging the *BIT* and the *BIT'* lines with 1's, and then asserting the *WORD* line for a given period. The *BIT* line gets set to the value stored in the cell via this procedure, and it may be used as the user desires. Typically the memory cells are placed vertically aligned, and common *BIT* and *BIT'* lines run through an entire column. Similarly the same *WORD* line runs through all the memory cells in the same row, and is asserted via a row decoder.

It is important to pay careful attention to the transistor sizes for the pass and pull-up and pull-down transistors in the cell. The pull-ups only have to offset the effects of leakage and thus are of minimum size. The pass transistors here were of size 2:4 and the pull-downs on the inverters were of size 2:4. These were found to be adequate given the timing criterion they had to satisfy.

4.2. Floor Plan of Memory Cells

The *BTT* has 64 entries. Each entry consists of 4, 9-bit words. Access is restricted to the selection of an entry to write to or read from. The memory is set up in 64 rows, each of which is accessed via a 6-bit address. There is a row decoder that decodes this 6-bit address. Each row has 36 memory cells, 20 of which are to the left of the row decoder, and the remaining 16 are to the right of it. The cells on the left store the lower order 5 bits of the four words, whereas the cells on the right store the higher order 4 bits of the four words. The circuit receives or provides the 4, 9-bit words serially over a 9-bit port.

4.3. Overall Operation

The *BIT* and *BIT'* lines are precharged high in preparation for a read cycle (this is the more frequent event) during the latter part of the packet cycle. Thus the first packet cycle during the operation of the circuit is called a precharge cycle and the circuit may not be put to useful operation during this precharge cycle.

4.3.1. Address Decoding

The address to the *BTT* arrives at local time 4 and is latched by the circuit at that time, irrespective of the packet cycle being a read or a write cycle. The address decoder needs to be precharged before being used. Thus it is necessary to precharge it until a few clock cycles before local time 4, and the precharge may begin a few cycles after local time 4.

4.3.2. Write Cycle

If the packet cycle is a write cycle, data arrives at local times 4, 5, 6, and 7. It is shifted in a serial-in-parallel-out *SIPO* shift register to facilitate placement of the 36-bits in memory. There are 9 such registers each of which has a length of 4 bits. The data is then latched by the the *SIPO* latches (these are just a set of clocked inverters). The data is then provided to the column drivers shown in the block diagram. These column drivers drive the *BIT* and *BIT'* lines to the appropriate values. Recall that the *BIT* and *BIT'* lines are precharged high in expectation of a read cycle. Thus it is necessary to discharge the *BIT* and *BIT'* lines to set them up to hold the values. On receipt of an input bit, the column driver drives the two lines to their appropriate values and continues driving them for 8 clock cycles. The pass transistors at the selected address are opened (the address decoder asserts the appropriate word line) 4 clock cycles after the column

driver start driving the lines, and the value on the *BIT* line is stored in the memory cell. The pass transistor is closed after another 4 clock cycles.

4.3.3. Read Cycle

If the packet cycle is a read cycle, then the *BIT* and *BIT'* lines are set up by the precharge portion of the previous packet cycle. The pass transistors are opened at local time 8 and kept open for 3 clock cycles by the address decoder asserting the word line corresponding to the requested address. The memory cells place their values on the *BIT* line, and these are latched downstream by the parallel-in-serial-out *PISO* latches. The latched values are then shifted out through 9, 4-bit *PISO* shift registers. The 9 outputs are placed on the downstream side *dd* of the circuit.

4.4. BTT Control and Timing

The input to the control and timing generator used locally in the *BTT* is provided in Figure 19. The output signals from this control and timing circuit are used to synchronize the operation of the components of the *BTT*. The packet cycle is 86 clock cycles long. The *BIT* and *BIT'* lines are precharged from local times 21 through 84. The address is latched at time 4, and the address decoder is precharged from times 6 through 20.

If the packet cycle is a write cycle, then the *BIT* and *BIT'* lines are discharged between time 4 and 7, and the input is latched at time 8 as soon as it has shifted through the *SIPO* registers. The column drivers drive this onto the *BIT* and *BIT'* lines between clock cycles 11 and 18. The pass transistors are opened at time 15, after the lines have charged up for 4 clock cycles. The pass transistors are closed at time 19 which allows enough time for the cell to acquire the correct value.

If the packet cycle is a read cycle, then the pass transistors are opened at the requested address between times 8 and 11, which allows enough time for the memory cells to discharge the *BIT* line if the stored value is a 0. The values on the *BIT* lines are latched downstream at local time 15, and shifted out appearing on the output lines at times 16, 17, 18, and 19.

4.5. SIPO and PISO shift registers

The *SIPO* and *PISO* shift registers are 4-bit long dynamic shift registers built from a chain of clock inverters. A single stage of the chain contains 2 clocked inverters with the first inverter clocked on $\phi 1$ and the second on $\phi 2$. During the interval between the two phases data is stored on the capacitance of the internal node.

The upper limit of operation is dependent on the delay through a clocked inverter and the time necessary to charge the internal nodes. This has been found to be about 125 *MHz*, and the lower limit is about 4.6 *KHz*.

4.6. SIPO and PISO Latches

These latches are just simple clocked inverters. The *PISO* latch on the downstream side acts as a sense amplifier for the corresponding *BIT* line. They do not need to drive the output lines powerfully because the cells on the downstream side are adjacent to the latches. Again the upper limit of operation is about 125 *MHz* and the lower limit is about

```
/* Input description for control and timing circuit for the BTT
*/

  condinps rwbar
  muxouts precharge discharge mil eval wordw wordr lddr lat addpre
  cntsize 7
  cntmax 90
  cntena - 86 <MAX>
  sigset - 21 precharge
  sigrst - 85 precharge
  sigset 0 4 discharge
  sigrst 0 8 discharge
  sigset 0 8 mil
  sigrst 0 9 mil
  sigset 0 11 eval
  sigrst 0 19 eval
  sigset 0 15 wordw
  sigrst 0 19 wordw
  sigset 1 8 wordr
  sigrst 1 11 wordr
  sigset 1 15 lddr
  sigrst 1 16 lddr
  sigset - 4 lat
  sigrst - 5 lat
  sigset - 6 addpre
  sigrst - 21 addpre
```

Figure 19. BTT Control & Timing Circuit Description

350 *KHz*.

4.7. Column Drivers

The column drivers have a 4:12 length to width ratio and are used to drive the *BIT* and *BIT'* lines of memory, each of which has a capacitance of about 1.8 *pF*. For an operating frequency of 50 *MHz*, four clock cycles allow 80 *ns* to charge the *BIT* and *BIT'* lines, which is more than adequate for the purpose.

4.8. Address Decoder and Drivers

The address decoders and drivers were obtained from the packet buffer chip described in [Ba88], and were found to be adequate for the purpose. They are allowed a minimum of 60 *ns* at a clock rate of 50 *MHz* to charge the word lines on a row high, which is quite conservative.

5. Testing Scheme

The previous chapters in this report have already dealt with the isolation and monitoring of various buses of interest. The testing scheme described here has to do with the simulation of the circuit representation. There are two major types of tests that may be performed. The event simulator *ESIM* allows a complete test of the functionality of the circuit without any consideration given to the real clock rate. The timing simulator *FACTS* allows us to monitor various signals in the circuit at some selected clock rate.

5.1. ESIM Simulation

The general approach taken here was to simulate functionally all components of the circuit, toggle all major lines in the internal buses and the memory bits, and simulate the overall functionality of the circuit from the pads. In general the circuits were provided sequences of zeros, walking and running ones, followed by sequences of zeros to ensure that all the bits in the circuit were toggled at least once. The current implementation of the simulator has a limitation on the vector size as a result of which the longest simulation run possible was 6 packet cycles.

5.1.1. Simulation of the Components

Each component in the circuit was tested via *ESIM* . The *BTT* is the only circuit that saves state information from one packet cycle to another. Thus this was simulated using a 6 packet cycle input deck in a precharge - write x - write y - read y - read x - read y sequence. Both *SSP* s were subjected to all four possible conditions that they deal with. These conditions could be enumerated as the processing of either a broadcast data packet, a *WBTT* packet, a *RBTT* packet, and a "nothing" packet which includes the point-to-point case. The global control and timing circuit generates all its output values in one packet cycle which is enough for a complete functional test. The rest of the components are minor and do not operate on a packet cycle. Indeed some of them are completely asynchronous with respect to the clock.

5.1.2. Simulation of the Overall System - Taps in Watch Mode

The most important task here is to pass the four major types of packets that the *BTC* encounters through it, as well as test the state information in the *BTT* within a span of six packet cycles. The *BTT* requires a precharge cycle before it is able to do anything useful. This necessitates the resetting of the *perr* signal via a *sr* signal at the end of the precharge cycle. All of these conditions were met using the following packet sequence.

- precharge cycle with *sr* at the at logical time 83
- *WBTT* packet with the address field = 000000
- *WBTT* packet with the address field = 111111
- *RBTT* packet with the address field = 111111
- broadcast packet with bcn = 000000
- point-to-point packet

The corresponding and necessary test is to perform a *RBTT* and broadcast translation on the same address. The addresses 000000 and 111111 toggle all the address bits and

inspire some confidence in the address decoding of the *BTT*.

The test access points were set up to watch *Tap* numbers 0, 1, and 2 during the three simulation runs dictated using the above policy, thus providing an exhaustive check of the test access system's monitoring capacity

5.1.3. Test Mode Simulation

In the test mode, the pads drive the test access points externally. This implies that the values of variables on the internal buses being controlled by the test access points will be those provided externally. It is important to ensure that the external data provided, while different and preferably the complement of the original data on the bus, does not interfere with the current function of the circuit. It is important to observe the effect of the new data on the next circuit component while in the test mode. The following set of tests was devised to satisfy all the constraints. The nodes being watched are not listed but all necessary buses were monitored during the tests and nothing untoward was observed. Each of the tests was performed with 3 packet cycles in a precharge, *WBTT* to address x, and broadcast packet sequence.

- Drive Buses A, B, and C with *Tap* numbers 0. This tests the *BTT* independently of the other circuitry.
- Drive Bus A with *Tap* number 1.
- Drive Bus B with *Tap* number 1.
- Drive Bus C with *Tap* number 2.
- Drive Bus A with *Tap* number 2, and Bus C with *Tap* number 1. It is possible to drive these two together without affecting the basic functionality of the circuit.

5.2. FACTS Timing Simulation

The *FACTS* simulator allows the user to specify real signals and use them to drive the inputs on a circuit block. A simulation which takes real effects such as capacitance and resistance into account is then performed on the circuit. The output from the simulator is a set of waveforms representing selected nodes in simulated time. Typically one would select a couple of the inputs and various different outputs of the circuit block. The output generated from this simulator has been found to be close to the output achieved for the chip itself and therefore is very useful.

5.2.1. Implementation Issues

The *FACTS* simulator is currently implemented for a MicroVax II. In the current implementation it is not realistic to attempt to simulate circuits with more than 10,000 transistors in them because they are likely to either crash the machine or take too much time to complete. Therefore each of the major components in the *BTC* was simulated independently.

Since *FACTS* is a realistic simulator, an attempt was made to provide realistic signals at the input lines, and to put a realistic load on the output lines. This may be achieved for the output lines by placing on them a load (a wire + the input to an inverter) close to the situation in which the circuit block is being used in *BTC*. The input signal

to each of the inputs on a circuit block depends on the output of the previous block, which is quantity that is not available yet. Therefore each of the input signals was simulated by running the signal through a pair of inverters before it enters the circuit block itself. The clock input signals were outputs of the clock drivers used in the circuit itself. The inputs to this setup circuitry are ideal signals. Figure 20 shows the block diagram for the parity generator circuit as used for the *FACTS* simulation.

5.2.2. Testing Performed on Blocks

All the simulations were performed at 40 MHz, i.e. with a clock period of 25 *ns*. The target for the actual chip is 25 MHz, and the simulated signals were strong enough at 40 MHz to be confident about running the chip at 25 MHz.

A list of the *FACTS* simulations performed appears below

- 3 runs for the *BTT*
- 1 run for the global *ctltim*
- 1 run for an 8 port *Tap tbacc* 8
- 1 run for a parity generator *pgen*

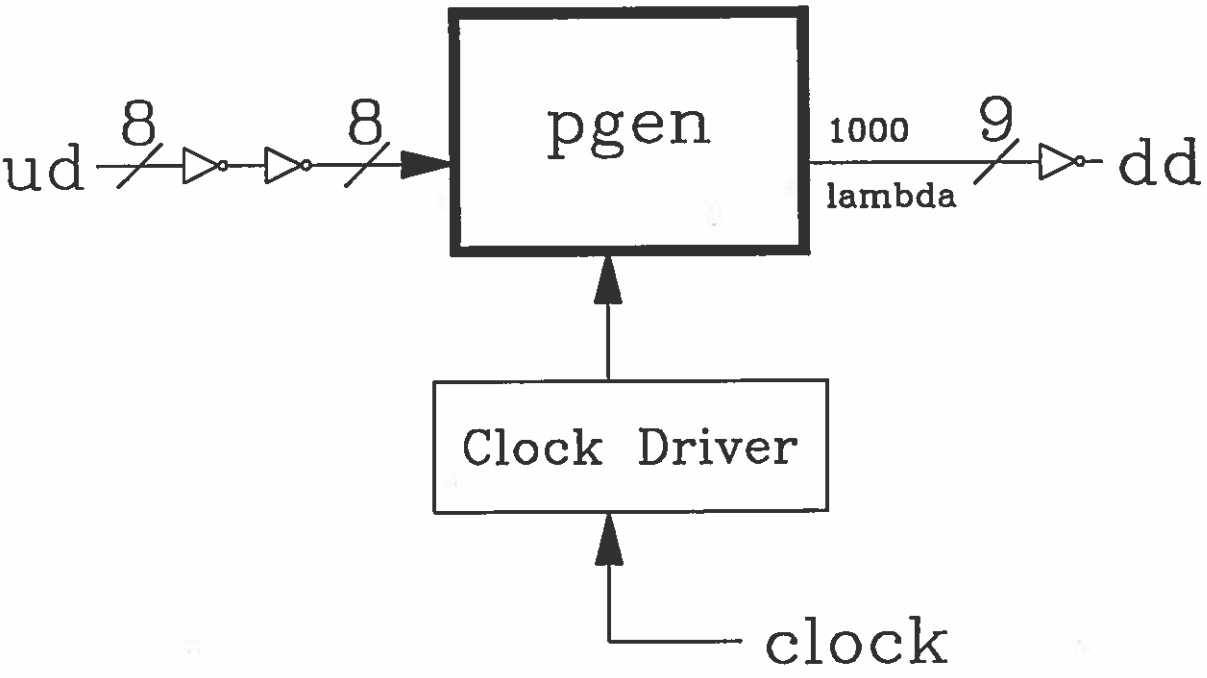

Figure 20. Parity generator Circuitry as simulated in FACTS

- 1 run for the input circuit *INC*
- 1 run for the output circuit *OUTC*

The *BTT* was simulated with half the memory plane removed. The control signals and the address decoder outputs are the only signals affected by this change. However the control signals either have a long time to rise or are driven through strong clock drivers and the additional capacitance will not affect them enough for them to fail. The address decoder provides very strong signals also and it was taken from a circuit with much larger memories where the chip actually worked, thus providing confidence in it also.

There are delay lines in the *SSP* s, and so it was not considered essential to simulate the delay line *delay*. Similarly the parity checker *pchk* looks almost exactly like the *pgen* and thus was not simulated. The pads are standard pads and have been rigorously tested previously.

5.2.3. Results

In general the results were satisfactory. The weakest signals at 40 MHz were the output signals from the *SSP* s, but they were strong enough to be latched by the following circuit. The *BTT* provided very clean signals as did the small circuits.

5.3. Preliminary Device Testing

MOSIS delivered 24 108-pin PGA devices in early October 1989. These devices are labeled *BTC* 0 through *BTC* 23 and were tested on a Tektronix LV500 system. The first observation was that there did not appear to have been a wiring mistake of any sort. The test access system worked correctly at 11 *MHz*, but not at 21 *MHz*. This was expected because speed of the test access system is limited by the large capacitance associated with each of the wires that make up the test bus.

The critical test was the validity of downstream data. It was found that 16 of the 24 devices had valid downstream data at an operating speed of 21 *MHz*. However there were zero devices that had valid downstream data at 25 *MHz*. The only errors that appeared at 25 *MHz*, in the downstream data (for those 16 devices) were in data read out of the *BTT*.

It was found that if data was written to the *BTT* at an operating speed higher than 21 *MHz*, then it was not possible to retrieve data from the memory. However, if the user wrote to the *BTT* at 21 *MHz*, and read data out at an operating speed of 25 *MHz*, then the output data was correct. Thus one would conclude that the *BTT* write cycle had a maximum operating speed of 21 *MHz*, and that this was the design bottleneck. Currently the *BTT* is being studied so that new devices would operate correctly at 25 *MHz*.

6. REFERENCES

[Ba88] Barrett, Neil, Design of a VLSI Packet Buffer, Washington University Computer Science Department, WUCS-88-32, St. Louis, 1988.

[Ro88] Robbert, George, A Circuit Generator for Synchonized Streams Processors, Master's Thesis, Washington University Computer Science Department, St. Louis, May 1988.

[Tu85] Turner, Jonathan S., Design of a Broadcast Packet Switching Network, Washington University Computer Science Department, WUCS-85-4, St. Louis, March 1985.

[Tu86] Turner, Jonathan S., Design of a Broadcast Packet Network, Proceedings of the IEEE Infocom'86, IEEE Computer Society, Los Angeles, April 1986, pp. 667-675. March 1985.

[WeEs85] Weste, Neil H. E., and Eshraghian, Kamran, Principles of CMOS VLSI Design, Addison-Wesley, Reading, Mass., 1985.

7. Appendix A - PINOUT

A description of the pin numbers on the chip packaging is provided here.

1 *Substrate Contact* - Always connected to pin 1 - for NWELL process this is connected to GND.

2 *Clocks* - ϕ 1 at pin 2, ϕ 2 at pin 3, ϕ 3 at pin 4.

3 *Packet Time* - *pt* at pin 5, which is responsible for signalling the start of packet cycle. Upstream data arrives 4 clock cycles after *pt*.

4 *Hard Reset* - *hrst* at pin 6, responsible for reseting the chip, clears all control circuitry and the parity error (*err*) signal

5 *Parity Signals* - *err* at pin 8, which signifies a parity error, *srst* at pin 7, which clears the *err* signal.

6 *Basic Chip Test* - *direct_in* at pin 25, *direct_out* at pin 26. The output at direct_out is equal to the input at direct_in. *invert_in* at pin 22, *invert_out* at pin 23. The output at *invert_out* is the inverse of the input at *invert_in*.

7 *Downstream Data* - Downstream data pins *dd_*0 at pin 24, *dd_*1 .. *dd_*8 at pins 31 through 38

8 *Power Pins* - *Vdd* at pins 39, 43, 93, & 97. *GND* at pins 40, 43, 94 & 96.

9 *Test Bus C* - *tbC*0 .. *tbC*3 at pins 51 through 54, *tbC*4 .. *tbC*9 at pins 56 through 61. These are bidirectional test bus pins connnected to test bus C.

10 *Direction Bit–BusC* - *tbCd* at pin 62 controls the direction of data on test bus C.
Normal Operation : *tbCd* = 1 - The direction of data flow is out of the chip and the data on the internal bus (determined by the *Tap* selected by *tbCs*0 and *tbCs* 1r) is being observed at the external pins connected to this test bus.
Test Mode : *tbCd* = 0 - The direction of data flow is into the chips, and the external pins are driving the said internal bus and replacing the data that appears on the downstream side of the bus.

11 *Address Pins* - Bus C - *tbCs*0 & *tbCs* 1 at pins 91 and 92 provide a 2 bit address that allows the user to select one of the 3 test access points connected to test bus C.

12 *Test Bus B* - *tbB*0 .. *tbB*9 at pins 63 through 72. These are bidirectional test bus pins connnected to test bus B.

13 *Direction Bit–BusB* - *tbBd* at pin 73 controls the direction of data on test bus B.
Normal Operation : *tbBd* = 1 - The direction of data flow is out of the chip and the data on the internal bus (selected by *tbBs*0 .. *tbBs* 1) is being observed at the external pins connected to this test bus.
Test Mode : *tbBd* = 0 - The direction of data flow is into the chips, and the external pins are driving the said internal bus and replacing the data that appears on the downstream side of the bus.

14 *Address Pins* - Bus B - *tbBs*0 & *tbBs* 1 at pins 89 and 90 provide a 2 bit address that allows the user to select one of the 2 test access points connected to test bus B.

15 *Test Bus A* - *tbA* 0 .. *tbA* 6 at pins 75 through 81, *tbA* 7 .. *tbA* 9 at pins 83 through 85. These are bidirectional test bus pins connnected to test bus A.

16 *Direction Bit–BusA* - *tbAd* at pin 86 controls the direction of data on test bus B.
Normal Operation : *tbAd* = 1 - The direction of flow is out of the chip and the data on the internal bus (selected by *tbAs* 0 .. *tbAs* 1) is being observed at the external pins connected to this test bus.
Test Mode : *tbAd* = 0 - The direction of flow is into the chips, and the external pins are driving the said internal bus and replacing the data that appears on the down-stream side of the bus.

17 *Address Pins* - Bus A - *tbAs* 0 & *tbAs* 1 at pins 87 and 88 provide a 2 bit address that allows the user to select one of the 3 test access points connected to test bus A.

18 *Upstream Data* - Upstream data pins *ud_* 0 ..*ud_* 8 at pins 107 through 99, i.e. *ud_* 0 at pin 107, *ud_* 1 at pin 106, *ud_* 2 at pin 105 etc.

8. Appendix B - PIN PACKAGING

	M	L	K	J	H	G	F	E	D	C	B	A
1		tbA6	tbA5	tbA2		tbB8	tbB5	tbB2	tbCd	tbC7	tbC5	
	(82)	(81)	(80)	(77)	(74)	(71)	(68)	(65)	(62)	(59)	(57)	(55)
2	tbA8	tbA7	tbA4	tbA1	tbBd	tbB7	tbB4	tbB1	tbC9	tbC6	tbC4	tbC3
	(84)	(83)	(79)	(76)	(73)	(70)	(67)	(64)	(61)	(58)	(56)	(54)
3	tbAd	tbA9	tbA3	tbA0	tbB9	tbB6	tbB3	tbB0	tbC8	tbC0	tbC1	tbC2
	(86)	(85)	(78)	(75)	(72)	(69)	(66)	(63)	(60)	(51)	(52)	(53)
4	tbBs0	tbAs1	tbAs0									
	(89)	(88)	(87)							(48)	(49)	(50)
5	tbCs1	tbCs0	tbBs1									
	(92)	(91)	(90)							(45)	(46)	(47)
6		gnd	vdd							gnd	vdd	
	(95)	(94)	(93)							(42)	(43)	(44)
7		vdd	gnd							vdd	gnd	
	(98)	(97)	(96)							(39)	(40)	(41)
8	ud6	ud7	ud8							dd6	dd7	dd8
	(101)	(100)	(99)							(36)	(37)	(38)
9	ud3	ud4	ud5							dd3	dd4	dd5
	(104)	(103)	(102)							(33)	(34)	(35)
10	ud0	ud1	ud2	hrst						dd0	dd1	dd2
	(107)	(106)	(105)	(6)	(9)	(12)	(15)	(18)	(21)	(24)	(31)	(32)
11		phi1	phi3	srst					inv_in	d_in		
	(108)	(2)	(4)	(7)	(10)	(13)	(16)	(19)	($\overline{22}$)	($\overline{25}$)	(29)	(30)
12	sub	phi2	tm4	err					inv_out	d_out		
	(1)	(3)	(5)	(8)	(11)	(14)	(17)	(20)	($\overline{23}$)	($\overline{26}$)	(27)	(28)
	M	L	K	J	H	G	F	E	D	C	B	A

9. Appendix C - BTC PHOTOMICROGRAPH