Design of an 8-BIT VLSI Packet Switch Element

Einir Valdimarsson

This paper describes the design of the Packet Switch Element Chip, one of the components of a high speed broadcast packet switching network. The chip is a 2x2 switch element, from which a switch fabric of arbitrary size can be constructed. It is currently being implemented in 2um CMOS technology.

Follow this and additional works at: https://openscholarship.wustl.edu/cse_research

Recommended Citation
DESIGN OF AN 8-BIT VLSI
PACKET SWITCH ELEMENT

Eimir Valdimarsson

WUCS-88-23

October 1988

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

Abstract
This paper describes the design of the Packet Switch Element Chip, one of the components of a high speed broadcast packet switching network. The chip is a $2 \times 2$ switch element, from which a switch fabric of arbitrary size can be constructed. This is a second version of the chip and it implements 8-bit data lines. It is currently being implemented in $2 \mu$m CMOS technology.

This work supported by Bell Communications Research, Bell Northern Research, Italtel SIT, NEC and National Science Foundation grant DCI-8600947.
Contents

1 Introduction ............................................. 3
   1.1 Packet Switch ......................................... 3

2 Chip Specification ....................................... 6
   2.1 Operation ............................................... 6
   2.2 PSE Interface .......................................... 7

3 Design of Major Functional Blocks ....................... 11
   3.1 Input Shift Register ................................... 11
   3.2 Input Control Circuit .................................. 12
   3.3 Buffer Shift Register .................................. 14
   3.4 Header Modification Circuit ......................... 15
   3.5 Timing Control Circuit ............................... 17
   3.6 Output Control Circuit ............................... 18

4 Implementation ........................................... 21
   4.1 Integrated Circuit Design ............................. 21
   4.2 Circuit Design Verification ........................... 21
   4.3 Packaging and Pin Assignment ........................ 21
   4.4 Testability features .................................... 21
List of Figures

1  Switch Module .................................................. 3
2  Packet Format .................................................... 4
3  Packet Switch Element Interface ................................. 7
4  Packet Switch Element Block Diagram ......................... 9
5  Input Circuit ..................................................... 10
6  Input Shift Register ............................................. 11
7  Input Control Circuit ........................................... 12
8  Header Register and Decoder .................................. 13
9  Buffer Shift Register ........................................... 14
10 Header Modification circuit .................................... 15
11 Timing Control Circuit .......................................... 17
12 Timing Circuit Timing ........................................... 18
13 Output Control Circuit .......................................... 19
14 Pin Assignment of the PSE Chip ................................ 22
15 Test circuitry in Input Control Circuit ......................... 23
16 Test Circuitry in Header Modification Circuit ............... 25
17 Chip block diagram ............................................. 39
1. Introduction

This paper describes the design of the Packet Switch Element Chip (PSE) for use within a broadcast packet switching network. The broadcast packet switching network is described in [Tu85] and [Tu86]. This is a second version of the chip implementing 8-bit data lines. The main difference from the first version [St88a] is in grant propagation and number of buffers.

1.1. Packet Switch

The overall structure of the prototype packet switch is shown in Figure 1. The switch module terminates 7 fiber optic links, and consists of five major components: the connection processor (CP), packet processors (PPs), copy network (CN), broadcast translation circuits (BTCs), and routing network (RN). The CN, BTCs, and RN are collectively known as the Switch Fabric (SF), and are constructed from banyan-interconnected PSEs. The PPs serve as buffers between the transmission

![Figure 1: Switch Module](image-url)
Figure 2: Packet Format

links and also perform the address translation required by routing. The RN routes the packets to the appropriate output link based on information encoded in the routing field (RF) of the packet. Since a banyan interconnection is self-routing, the PSEs need only examine a single address bit to select the proper output port. The CN replicates broadcast packets to multiple output links based on information in the RF, indicating the fanout of the packet. Point-to-point packets are arbitrarily routed through the CN. The distribution network (DN), not present in the prototype, will randomly distribute packets before the RN to avoid congestion occurring from persistent poor traffic patterns (resulting, for example, from communities of interest).

The format of the packets processed by the PSE is shown in Figure 2. The packet is organized as a sequence of 8-bit words and an odd parity bit. Each packet contains exactly 80 words, the first five of which constitute the packet header. Only the first four words, the routing field (RF), are used by the PSE. The meaning of the fields are:

- **Routing Control (RC)**. These three bits determine how the packet is processed by the switch elements.
  - 000 signifies an empty packet slot,
  - 001 signifies a normal point-to-point data packet,
  - 010 signifies a normal broadcast packet,
  - 100 signifies a test packet.
- **Operation (op).** This field specifies which of several control operations is to be performed for this packet. Not used by the PSE.

- **Fanout (FAN).** If RC = 010, the second word of the packet is taken to be the fanout, that is the number of switch fabric output ports that require copies of the packet.

- **Link Number (LN).** If RC = 001, the second word of the packet is taken to be the number of outgoing link to which the packet should be delivered.

- **Broadcast Channel Number (BCN).** If RC = 010, the third and fourth words of the packet are taken to be the broadcast channel number. All packets within a particular multi-point channel have the same broadcast channel number. Only 256 distinct BCN are recognized.

- **Internal Logical Channel Number (ILCN).** If RC = 001, the third and fourth words of the packet are taken to be the internal logical channel number. This will become the external logical channel number when the packet exits the switch module.

- **Specific Path Specification.** If RC = 100, the next three words specify output ports for each of the three networks. The packet will routed through each of these.

- **Source (SRC).** This word identifies which switch fabric input port the packet came from, and is not used by the PSE.

- **Information (i).** This field normally contains user information. In the case of control packets, additional control information may be placed here.
2. Chip Specification

2.1. Operation

A single PSE circuit is used within the routing, copy, and distribution networks that comprise the switch fabric. PSE routing decisions are based on the operation mode (OM) and packet type (RT in the packet header). The operation modes are:

- *Route* — Route all packets based on the appropriate bit of the LN field of the packet header.
- *Copy* — Replicate packets based on the FAN field of the packet header.
- *Distribute* — Distribute packets arbitrarily and uniformly between outputs.

The packet types are:

- *Point-to-point* — Use LN to route packets only for route operation mode.
- *Broadcast* — Use LN to route packets only for copy operation mode.
- *Specific Path* — Use LN to route packets in all operation modes.

When arbitrary routing decisions can be made, the following policies are followed:

- Ties among input ports for the same output port are broken such that the last input port favored loses, to avoid individual starvation.
- Packets that can be routed to either output are uniformly distributed between output ports based on an output toggle. This includes all packets in the distribution network, and point-to-point and unreplicated broadcast packets in the copy network.
- Packets requiring both output ports in the copy network are favored over packets needing only one output.
- Packets requiring a specific output port are favored over packets requiring either output.

All packets are routed in the RN based on the LN. The PSES are interconnected to form a banyan network, which is self-routing. The routing algorithm is simply to route to output port 0 or 1, depending on whether the LN bit corresponding to the switch element stage number (SN) is 0 or 1.

Broadcast packets are replicated in the CN such that the number of packets leaving the CN is equal to the original FAN of the packet. Each PSE makes a decision to copy if the FAN is greater than half the number of reachable outputs, specifically if FAN > 2^SN. The FAN of each replicated packet is adjusted to reflect the number of copies that must be subsequently made in the CN. This copy algorithm delays packet replication as long as possible to reduce congestion in the CN. Point-to-point packets, and broadcast packets not requiring replication in a particular PSE, are routed based on the distribution algorithm described above.

Packets in the distribution network are randomly distributed among the outputs of each PSE. This is accomplished by toggling a favored output bit for each input port, and thus distributing packets evenly across the outputs of the switch element.

Each PSE operates on the basis of a packet cycle, indicating the timing of packet arrival. A packet cycle is initiated by packet-time going high, indicating that data is present on the upstream data
lines at the input to the PSE. The Packets are either buffered, or if possible cut-through directly to the appropriate output port. Each input is capable of fully buffering two 80 word (8 bits + parity) packets. If a packet is cut through, it appears on the downstream data lines at the appropriate output of the PSE twenty-four clock cycles later. Successive words of the packet are received/transmitted every clock cycle.

Grants are propagated backward from a node to its upstream neighbour, indicating that switch element stage will be able to accept packets (due to an empty buffer slot).

2.2. PSE Interface

The external interface signals of the PSE are shown in Figure 3 and are described briefly below. Signals incorporating multiple lines are represented collectively by square brackets, e.g. x[210] refers to x2, x1, x0, x[ab][10] refers to xa1, xa0, xb0, xb1, and x[10][10]' refers to x1, x0, x̅1, x̅0.
PACKET SWITCH ELEMENT

- **Upstream data leads (ud[876543210] [AB]).** Incoming data from upstream switch elements; 8-bits data plus odd parity for each input side of switch (a,b). Parity is upstream data lead ud0 [AB].

- **Downstream data leads (dd[876543210] [10]).** Outgoing data to downstream switch elements; 8-bits data plus odd parity for each output port of switch (0,1).

- **Upstream grants (ug[AB]).** Grant signals to upstream neighbors, granting permission to transmit a packet during the next packet cycle.

- **Downstream grants (dg[10]).** Grant signals from downstream neighbors, indicating permission has been given to transmit a packet on the corresponding output port during the next packet cycle.

- **Packet-time (pt).** Goes high when first word of a packet is present on the ud leads.

- **Operation mode (om[10]).** Switch element operation mode: 01 = RN, 10 = DN, 11 = CN.

- **Stage number (sn[210]).** Indicates the switch network stage of which the PSE is a part. The furthest downstream column of switch elements is stage 0.

- **Rotate routing field (rrf).** Causes the PSE to rotate the second through fourth words of test packets.

- **Error flag (err).** Report errors, including parity errors and packet header validity errors.

- **Hard reset (hrst).** Initialize internal control registers and reset err.

- **Soft reset (srst).** Reset error flag err.

- **Clock (ϕ₁, ϕ₂, ϕ₃).** Two-phase, symmetric, non-overlapping clock (ϕ₁, ϕ₂) and third phase (ϕ₃) overlapping ϕ₁ used in timing control circuit.

- **Power and Ground (Vdd, Gnd).**
The PSE is a buffered 2-input, 2-output switch with broadcast capability. The main data path is symmetric with respect to each side of the switch element (a,b) and each output (0,1). Control and arbitration logic is shared across both halves of the switch. A block diagram is shown in Figure 4.

The major components of the switch element are:

- **Output Control Circuit (OCC)** — The OCC arbitrates access to the two output ports, based on the downstream grant signals and port requests received from the input port circuits. The port requests are given in the form of three bit request vectors, \( r_{[01]} \) and \( r_{[01]} \). The response is given in the form of two bit enable vectors \( en_{[01]} \) and \( en_{[01]} \). It also keeps track of last favored input and output.

- **Timing Circuit (TC)** — This circuit generates signals of the form \( t_i \) and \( t_{ij} \), for various values of \( i,j \). Signal \( t_i \) is high during the \( i \)th clock period. Signal \( t_{ij} \) is high during \( t_i \) and stays high through \( t_j \). These signals are used as basic control signals for the rest of the PSE.

- **Input Circuit (ICP, IPC) — There is one input circuit for each input port. Each IPC includes two buffers large enough to hold a single packet, plus control circuitry to extract information from the packet header, generate the request vector for the OCC and use the resulting enable vector to make decisions on the disposition of the packets. It also modifies
the packet header when necessary. The structure of the input circuit is shown in Figure 5 and is explained below.

- **Input Shift Register** — The two input shift registers (ISRP, one per switch input) are 21 stage 9-bit static shift registers with an output tap after the first two stages. Packets are shifted into each input shift register from the corresponding upstream data lines. The first two stages of shift delay are for synchronization with pt. Parity is checked after the first two shift stages. The remaining 19 stages allow sufficient time for control and routing decisions to be made by packet header and output control circuit.

- **Input Control Circuit** — The ICC controls the flow of packets through the input circuit. It extracts and decodes header information from incoming packets and stores the decoded information for packets stored in the packet buffers. Using this information, it requests output ports from the OCC and based on the results, controls the flow of packets through the IPC. It also generates the upstream grant signals. Invalid header information sets an error flag, and forces the packet type to point-to-point (so that bad packets are not broadcast through the fabric).

- **Packet Buffer and Cut-Through Path** — The packet buffer shift registers (BSR, two per switch input) are 80 stage 9-bit static shift registers. Based on the availability of requested outputs, a packet is either shifted into a packet buffer, or cut through directly to an output port. A 9-bit 3-to-1 multiplexer selects one of the buffer or cut-through paths to the output routing logic.

- **Header Modification Circuit** — This component (HMC makes minor changes to the header as specified by the ICC. If the test bit is asserted, words 1-3 of the routing field are rotated, with word 1 becoming word 3, word 2 becoming 1 and word 3 becoming word 2. If the copy bit is asserted the packet is sent to both output ports and the fanout fields of the copies are modified. The low order bit of the bcn determines which copy gets the "extra" copy when the fanout value is odd. The en[01] signals control the passage of data onto the output links, with en0 enabling output 0 and en1 enabling output 1.
3. Design of Major Functional Blocks

This section provides a more detailed description of each of the functional blocks described previously.

3.1. Input Shift Register

The input shift register logic is shown in Figure 6.

The interface signals to the input shift register are:

- **Inputs**
  - *Upstream Data* (di[876543210]) – The 9-bit word input from the di inputs to the PSE.
  - *Clock* (Phi1, Phi2) – Shift register clocking.

- **Outputs**
  - *Data* (do[876543210]) – The 9-bit output from the shift register; input data delayed by 21 clock cycles.
  - *Header Data* (d[76543210]) – The 8-bit output from the first two stage of the shift register.
  - *Parity Check* (pe) – Outputs indicating odd parity error of data on the d lines.

The input shift register ISRP consists of three major components: a two stage 9-bit shift register, a parity check circuit, a nineteen stage 9-bit shift register and a clock driver.

The two stage shift register shifts 9-bit packet data in from the upstream data (di) inputs of the PSE, allowing for synchronization with the beginning of a packet cycle. The output of this stage is passed to the input control circuit block and to the later shift register stages.
The parity check circuit PAR is logically an EXCLUSIVE-OR tree to determine the parity of the incoming data. Output (pe) are passed to the ICC and ERROR to take appropriate action in case of a parity error. Parity is not restored if in error, but an error will cause the error output (err) of the PSE to go high.

The remaining 19 stages of shift allow the input control circuit circuit and output control circuit sufficient time to make control decisions that affect routing of the packet, before the packet proceeds too far into the PSE data path. These decisions include whether the packet is to be buffered or cut through, if any header modification is to occur, and to which output port the packet is destined.

3.2. Input Control Circuit

The input control circuit is shown in Figure 7.

The interface signals to the ICC are:
Figure 8: Header Register and Decoder

- **Inputs**
  - *Header Data* \(d1[76543210]\) – The 8-bit word from the output of the second stage of the input shift register, which is selectively loaded into the appropriate header registers.
  - *Header Register Load* \(ld_0, ld_1, ld_2, ld_3\) – The load control lines to load the header FA, RC and BCN into the corresponding header registers.
  - *Operation Mode* \(om[10]\) – The switch network operation mode within the switch fabric.
  - *Stage Number* \(sn[210]\) – The stage number of which this PSE is a part.
  - *Routing Control Parity Error* \(pe\) – Parity of RC was invalid.

- **Outputs**
  - *Output Port Request* \(ro[NO1]\) – Switch element output port(s) requested: 000=(no port), 100=(either port), 101=(port 1), 110=(port 0), 111=(both ports).
  - *Routing Control Bad* \(rcbad\) – Invalid RC, unknown packet type.
  - *Upstream Grant* \(ro9\) – Upstream Grant signal.
  - *Select Buffer* \(xo[56]\) – Select buffer BSR0, BSR1, shift signal for buffers.
  - *HMC control signals* \(ro[0123478]\) – Control signals for HMC: \(en[01]\), copy, bcn, test, bsel[01].

The header logic block HEADER consists of three major components: the header registers, the header request logic, and the header register multiplexor.

The header registers HDRREGS are four 8-bit registers (parity is not needed), which store the contents of the two header words RC, FA and also one bit of BCN. The four registers share a common input data bus, which comes from the input shift register ISRP. The load inputs are successively clocked by the timing control circuit TCC, as the header is shifted through ISRP.

The header decoder logic HDRDEC examines the relevant contents of the header registers and determines what output ports the packet requires. It includes the circuit HDRRC that examines the
and determines the packet type (nil, point, bdcst, test), as well as indicating if the RC is invalid (rcbad), and determining the high order request bit (rN), i.e., if any request is to be made. The circuit HDRFA examines the FA. The specific output port requested (req(1)) is determined by examining the SNth bit of the ADR in the case of a point-to-point packet. The relation of the FAN to the SN (eq2sn, lit2sn) is determined for broadcast packets (FAN = 2SN, FAN < 2SN). The HDRM circuit decodes the operation mode of the switch element. The HDRPORT circuit uses the outputs of HDRRC, HDRFA, and HDRM to determine the specific port request bits r[10]. The HDRDEC also computes the r[N], copy and test signals for incoming packets.

A PLA provides overall control of the ICC. If requesting ports are not available the decoded header information will be loaded into buffer control register BCR0, BCR1, corresponding to the buffer shift registers. Each BCR has six data inputs and six tristate data outputs. In addition the incoming header information signals have a tri-state output. The BCRs have two control inputs, controlled by PLA. Only one of the BCRs or the incoming vector can be enabled at a time, to provide the request vector to the OCC. The PLA also generates the upstream grant signals. Specification for the PLA is presented in appendix B.

3.3. Buffer Shift Register

The buffer shift register is shown in Figure 9.

The interface signals to the buffer shift register are:

- **Inputs**
  - *Data In* (di[876543210]) – The 9-bit input from ISRP.
  - *Shift Data* (s) – Shift data in the buffer/store data in the buffer.
  - *Clock* (phi1, phi2) – Shift register clocking.

- **Outputs**
- Data Out ($do[876543210]$) - The 9-bit word output from the buffer.

The packet buffer BSR is based on an 80 stage 9-bit shift register. The shift register additionally contains very large line drivers (CLKBUF) to drive the shift register clock lines, and control logic (CLKSED) to qualify shift and hold signals with phi1. Those signals shift or hold the packet in the buffer along with phi2.

3.4. Header Modification Circuit

The output header modification circuit is shown in Figure 10.

The interface signals to the header modification circuit are:

- Inputs
- **Data Input (d0[876543210])** – Packet data proceeding to one or both output ports, which has passed through the BSR0.
- **Data Input (d1[876543210])** – Packet data proceeding to one or both output ports, which has gone through cut-through path.
- **Data Input (d2[876543210])** – Packet data proceeding to one or both output ports, which has passed through the BSR1.
- **Buffer Select (bse1[10])** – Selection of path through BSR0, BSR1 or the cut-through path.
- **Output Enable (en[10])** – Enables the packet data to the appropriate output port.
- **Output Data from Other Port (op[876543210])** – Packet data from the output routing logic of the other half of the PSE.
- **Multiplexor Control (copy bcn test t1 t2)** – Signals used to control the multiplexors used for changing the header fields.

**Outputs**

- **Downstream Data (do[876543210])** – The 9-bit output to the dd outputs of the PSE.
- **Output Data to Other Half (tp[876543210])** – Packet data to the output routing logic of the other half of the PSE.

The header modification block HMC consists of several components: the data path multiplexor, the delay shift register D, the increment divide by two INCDIV the output multiplexor, and the output data latch.

The header modification circuit HMC performs modification of FAN for replicated broadcast packets. It consists of a PLA which increments and (integer) divides FAN by two and par calculating parity, and an EXCLUSIVE-OR gate (DIV) to maintain parity for a simple divide by two. The usual data path goes through the delaying shift register (d) and through the mainpath multiplexor which selects the BSRs or the cut-through path.

The output multiplexor OUTMUX selects the appropriate data path to proceed to switch port outputs, based on the output select from the control logic. The paths to be selected are:

- delayed data from BSR, the normal path for packet data
- data divided by two for FAN of replicated packets
- data incremented and divided by two for FAN of replicated packets
- undelayed data from MAINPATH, used only during test packet header rotation, controlled by the rrf PSE input

Two 9-bit 4-to-1 multiplexors are used, enabled such that during FAN modification of replicated packets, one output will receive the packet with FAN divided by two, and the other output the packet with FAN incremented and divided by two. The BON of the header controls this. In particular, if BON is odd, output port 0 will receive the packet with FAN simply divided by two, if BON is even it will be port 1. This allows the determination of which copy of a packet will reach a particular output port, which is required by the BTC to perform the correct channel translation on each copy.

Input from one multiplexor in OUTMUX is passed, if enabled, through to the downstream data register OUT. Input from the other multiplexor is output (tp) to the other half of the switch, destined for the corresponding output. Packet data from this switch half is combined (ored) with packet...
data from the other switch half (op) destined for this output. Only one of the outputs can have a path enabled to a given output port, so the enabling and combination of the data path behaves as a 2 x 2 crosspoint switch. The output is latched in a final two register stages (OUT), to be shifted out to the downstream data leads (dd) of the PSE.

3.5. Timing Control Circuit

The switch element timing control circuit is shown in Figure 11.

The interface signals to the timing control circuit are:

- **Inputs**
  - *Clock (phi1, phi2, phi3)* – Clock signal used by TCC.
  - *Reset (rst)* – Reset signal.
- **Outputs**
  - *Control Outputs (t1, t2, t3, t4, t16)* – Control outputs qualified with phi1.
- Control Outputs (t19, t20, t22, trot, tshift) - Control outputs lasting one clock cycle starting at phi2. except for tshift which is the time when data can be shifted into or out of the buffers and trot which lasts two clock cycles.

The timing control circuit consists of a counter, to count the clock cycle in the beginning of a packet cycle, and a dynamic PLA to decode the various times and guard conditions. The outputs of the counter are fed into the decoding PLA. The PLA outputs indicate the various clock times. The PLA outputs are formed by oring together some product terms. They are then connected to a set of R-S flip-flops used to generate the enables lines. The enable lines are the buffered outputs of the flip-flops. The reset signal is latched into a flip-flop and the circuit is reset until the next packet time arrives. The TCC was generated using a timing circuit generator designed by George Robbert [Go88].

The timing diagram for the timing circuit is shown in figure 12.

3.6. Output Control Circuit

The output control circuit is shown in Figure 13.

The interface signals to the output control circuit are:

- Inputs
  - Output Port Requests (r[NO1][AB]) - The 3-bit request vectors from the header logic of each switch side (ra, rb). The interpretation of the vector is: 000=(no port), 100=(either port), 110=(port 0), 101=(port 1), 111=(both ports).
Figure 13: Output Control Circuit
- **Downstream Grants** (dg[10]) – Grant signals from downstream neighbors, indicating that permission has been given to transmit a packet on the corresponding dd port during the next packet cycle.

- **Latch Tiebreaker signals** (t20) – A control signal indicating that the uI, uO inputs should be latched for the next packet cycle.

- **Clock** (phi1, phi2) – Clock signals used for the tiebreak latch.

- **Outputs**

  - **Output Enables** (e[10][AB]) – Switch element output port enable signals to each output side of the switch. The interpretation of the vector is: 00=(no port), 01=(port 0), 10=(port 1), 11=(both ports).

The output control circuit block OCC consists of a OCCPLA to generate the outputs, along with random logic to control the latch for the tiebreaker signals.

The OCCPLA uses the request vectors r[NO1][AB] and additional state vectors uI, uO to generate the output enables e[ab].

The additional state vectors is the last-used toggle (uI, uO. This is a two bit code indicating the most recently used switch input (uI=0:a, uI=1:b) and output (uO=0, uO=1). These allow the routing policies of arbitrary input tiebreaking, and uniform output distribution (under the proper circumstances) to be followed. During each packet cycle, new values are generated (newu[10]) based on the inputs and outputs used.

The full specification for enable generation is presented in appendix A.
4. Implementation

This section discusses the implementation of the PSE as a 2μm CMOS chip.

4.1. Integrated Circuit Design

The PSE chip was designed using a set of CAD tools from the University of California at Berkeley, described in [ScMa86]. The circuit layout tool used is MAGIC, which is a graphical layout editor providing a hierarchical cell structure. Technology-based design rules are checked by MAGIC, and are based on a Manhattan Mead-Conway design style with λ=1.0μm [MeCo80]. This results in a minimum line width of 2μm. Chip fabrication technology is a bulk p-well CMOS process, with facilities provided by MOSIS [US08]. Various references used during the design of the PSE are listed in the bibliography.

4.2. Circuit Design Verification

The PSE functional design was executed in a top-down hierarchical manner, but the chip was implemented bottom-up. As each cell was designed, it was simulated before incorporation into higher level cells. The various tools which were available for PSE simulation will be briefly described.

Circuit-level simulation was performed using FACTS [McNa]. Due to the computational intensity of these simulations, it was only used up to the major blocks and not on the whole chip except for the clock signals and major control signals. All low-level components were simulated in this manner, particularly the basic flip-flops, shift register cells, and clock line drivers. FACTS simulations gave us a good way to do timing analysis.

Logic-level simulation was executed using ESIM [ScMa86], which is efficient enough to allow simulation of all PSE logic functions, to the top level.

VESIM, a front-end interface to ESIM, was written by Tony Mazraani, and allows test vector input and simulation results to be printed in a vertical format. This enables test vectors considerably longer than 80 to be easily specified, and results interpreted, since the width of a terminal or print line is not a limit. It also gives a good way to compare ESIM output vectors with DAS output vectors, since the chip will be tested with the Tektronics DAS 9100 digital analysis system.

4.3. Packaging and Pin Assignment

The chip is implemented in a 108 pin grid array package. This package is Kyocera KD-P87938-A or equivalent with a 0.450" cavity (topside). The package is 1.200" square with the pins arranged on an 12 x 12 pin grid at 0.1" centers. There are three full rows of pins around the outside. Figure 14 depicts the bonding pad to pin connectivity as viewed from the top of the package (or the top of the socket onto which the package is plugged), with the index at the upper right corner. The columns of the grid are labeled alphabetically and the rows numerically. The signal names associated with the pins are also shown. The full description of pins can be found in Appendix C.
<table>
<thead>
<tr>
<th></th>
<th>M</th>
<th>L</th>
<th>K</th>
<th>J</th>
<th>H</th>
<th>G</th>
<th>F</th>
<th>E</th>
<th>D</th>
<th>C</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sn2</td>
<td>sn1</td>
<td>sn0</td>
<td>tm1</td>
<td>tdA1</td>
<td>tdA3</td>
<td>tdA5</td>
<td>tdA7</td>
<td>tbo1A</td>
<td>tbi1</td>
<td>tpo1A</td>
<td>wdt</td>
</tr>
<tr>
<td></td>
<td>(82)</td>
<td>(81)</td>
<td>(80)</td>
<td>(77)</td>
<td>(74)</td>
<td>(71)</td>
<td>(68)</td>
<td>(65)</td>
<td>(62)</td>
<td>(59)</td>
<td>(57)</td>
<td>(55)</td>
</tr>
<tr>
<td>2</td>
<td>phi1</td>
<td>hrst</td>
<td>phi3</td>
<td>tm0</td>
<td>tdA0</td>
<td>tdA2</td>
<td>tdA4</td>
<td>tdA6</td>
<td>tboQA</td>
<td>tpoOA</td>
<td>tsten</td>
<td>rrf</td>
</tr>
<tr>
<td></td>
<td>(84)</td>
<td>(83)</td>
<td>(79)</td>
<td>(76)</td>
<td>(73)</td>
<td>(70)</td>
<td>(67)</td>
<td>(64)</td>
<td>(61)</td>
<td>(58)</td>
<td>(56)</td>
<td>(54)</td>
</tr>
<tr>
<td>3</td>
<td>udA2</td>
<td>udA1</td>
<td>udA0</td>
<td>ten</td>
<td>tso10</td>
<td>tsi1</td>
<td>tso1A</td>
<td>tsoOA</td>
<td>tbiO</td>
<td>dd02</td>
<td>dd01</td>
<td>dd00</td>
</tr>
<tr>
<td></td>
<td>(86)</td>
<td>(85)</td>
<td>(78)</td>
<td>(75)</td>
<td>(72)</td>
<td>(69)</td>
<td>(66)</td>
<td>(63)</td>
<td>(60)</td>
<td>(51)</td>
<td>(52)</td>
<td>(53)</td>
</tr>
<tr>
<td>4</td>
<td>udA5</td>
<td>udA4</td>
<td>udA3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>dd05</td>
<td>dd04</td>
<td>dd03</td>
</tr>
<tr>
<td></td>
<td>(89)</td>
<td>(88)</td>
<td>(87)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(48)</td>
<td>(49)</td>
<td>(50)</td>
</tr>
<tr>
<td>5</td>
<td>udA8</td>
<td>udA7</td>
<td>udA6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>dd08</td>
<td>dd07</td>
<td>dd06</td>
</tr>
<tr>
<td></td>
<td>(92)</td>
<td>(91)</td>
<td>(90)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(45)</td>
<td>(46)</td>
<td>(47)</td>
</tr>
<tr>
<td>6</td>
<td>ugA</td>
<td>gnd</td>
<td>vdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>gnd</td>
<td>vdd</td>
</tr>
<tr>
<td></td>
<td>(95)</td>
<td>(94)</td>
<td>(93)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(42)</td>
<td>(43)</td>
<td>(44)</td>
</tr>
<tr>
<td>7</td>
<td>ugB</td>
<td>vdd</td>
<td>gnd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>vdd</td>
<td>gnd</td>
</tr>
<tr>
<td></td>
<td>(98)</td>
<td>(97)</td>
<td>(96)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(39)</td>
<td>(40)</td>
<td>(41)</td>
</tr>
<tr>
<td>8</td>
<td>udB6</td>
<td>udB7</td>
<td>udB8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>dd16</td>
<td>dd17</td>
<td>dd18</td>
</tr>
<tr>
<td></td>
<td>(101)</td>
<td>(100)</td>
<td>(99)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(36)</td>
<td>(37)</td>
<td>(38)</td>
</tr>
<tr>
<td>9</td>
<td>udB3</td>
<td>udB4</td>
<td>udB5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>dd13</td>
<td>dd14</td>
<td>dd15</td>
</tr>
<tr>
<td></td>
<td>(104)</td>
<td>(103)</td>
<td>(102)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(33)</td>
<td>(34)</td>
<td>(35)</td>
</tr>
<tr>
<td>10</td>
<td>udB0</td>
<td>udB1</td>
<td>udB2</td>
<td>tdB</td>
<td>tsolB</td>
<td>tsoOB</td>
<td>tseiO</td>
<td>tsei1</td>
<td>tbo1B</td>
<td>dd10</td>
<td>dd11</td>
<td>dd12</td>
</tr>
<tr>
<td>11</td>
<td>phi2</td>
<td>omO</td>
<td>om1</td>
<td>tso</td>
<td>tdbO</td>
<td>tdb2</td>
<td>tdb4</td>
<td>tdb6</td>
<td>tpo1B</td>
<td>tpioB</td>
<td>tpi1A</td>
<td>tboOB</td>
</tr>
<tr>
<td>12</td>
<td>sub</td>
<td>pt</td>
<td>srst</td>
<td>tsi</td>
<td>tdb1</td>
<td>tdb3</td>
<td>tdb5</td>
<td>tdb7</td>
<td>tpoOB</td>
<td>tpi1B</td>
<td>tpi1OA</td>
<td>err</td>
</tr>
</tbody>
</table>

Figure 14: Pin Assignment of the PSE Chip
4.4. Testability features

A very important part of the design was testability considerations. This included pins for observation of critical signals and circuit design to disable parts of the circuit and control the rest externally. To obtain this, almost half of the 108 pins of the chip are used for testing purposes.

The main concern was being able to access the ICC. The ICC with the additional testing circuitry is shown in Figure 15.

A 8 x 4 : 1 multiplexor makes it possible to observe the header registers and the input and output from the PLA which has the overall control of the packet flow through the PSE. Also the request vector \( r_{[NO1]} \) is observed. The multiplexor output is \( td_{[AB]}[7654321] \), and is enabled by the following:
Figure 16: Test Circuitry in Header Modification Circuit:

- \( \text{tm}[10] = 00 \) — Output of \( \text{pl}a \).
- \( \text{tm}[10] = 01 \) — Input of \( \text{pl}a \) and the request vector. \( \text{td}[765] = x[001] \).
- \( \text{tm}[10] = 10 \) — Output of \( \text{fan} \) register.
- \( \text{tm}[10] = 11 \) — Output of \( \text{rc} \) register.

An additional test register is used to bypass the header logic decoder and load the request vector straight in. A \( 5 \times 2 : 1 \) multiplexor is used to select either the decoder or the test register. \( \text{ten} = 0 \) selects the decoder and \( \text{ten} = 1 \) selects the test register. Furthermore there is also included the possibility to serial load the buffer control registers \( \text{BCR} \). If signal pin \( \text{td} \) is high the data on \( \text{ts}i \) is serially loaded and \( \text{ts}o \) is the output serial data. The \( \text{BCR} \) for the two ports are serially connected in the following order from the input \( \text{ts}i \): \( \text{BSR}0_B, \text{BSR}1_B, \text{BSR}0_A, \text{BSR}1_A \).

The shift signal for the buffer shift registers can be observed with \( \text{ts}o[10][AB] \) and the buffers can also be controlled externally with \( \text{ts}i0 \) for \( \text{BSR}0 \) for both ports and \( \text{ts}ii \) for \( \text{BSR}1 \) for both.
ports. Similarly the select signal of the mainpath multiplexor can be observed at tbo[10][AB] and controlled externally with tbi[10], same signal controls both ports. The enable signals to enable the ports can be observed at tpo[10][AB] and controlled externally with tpi[10][AB]. The multiplexors in HMC can also be controlled with tmei[10] as follows:

- tmei[10] = 00 — select divide by two
- tmei[10] = 01 — select increment and divide by two
- tmei[10] = 10 — select delayed path
- tmei[10] = 11 — select undelayed path

These testing features were accomplished by adding some multiplexors in ICC and HMC, the circuit of HMC is shown in Figure 16. The testing input is enabled when tsten is asserted high.
References


[usca] MOSIS User's Manual, University of Southern California Information Sciences Institute, Marina Del Rey, Calif., (no date)

[uscbl] MIT-SC3U-PADS.CIF, University of Southern California Information Sciences Institute, Marina Del Rey, Calif., (no date)


A. Packet Switch Element Routing Policy

This section documents the PSE routing policy in detail. First, all of the required inputs, outputs, and internal state variables are described. Then a set of tables describing the policy implementation is given. A general description of the routing policy is given in §2.1 of this technical report.

The tables are used as input to the MPLA program to generate the PLA ENGEN, which is used in the output enable circuit ENABLE to generate the e vectors.

An entry of “x” for an input variable indicates a “don’t care” condition. An entry of “−” for an output variable indicates that its output state will not change.
A.1. State variable definition

Output request vectors:

ra \[r_a^{[10]}\] three bit code specifying output ports requested by input a
rb \[r_b^{[10]}\] three bit code specifying output ports requested by input b
r \[r^{[10]}\] three bit code specifying output ports requested by an input

rN specifies that an output port is needed
r1 specifies output port 1 is needed
r0 specifies output port 0 is needed

0XX specifies no ports requested
100 specifies either (one) output port needed
101 specifies output port 0 needed
110 specifies output port 1 needed
111 specifies both output ports needed

Output enable vectors:

ea \[e_a^{[10]}\] two bit code specifying output ports enabled for input a
eb \[e_b^{[10]}\] two bit code specifying output ports enabled for input b
e \[e^{[10]}\] two bit code specifying output ports enabled for an input

e1 specifies output port 1 will be enabled
e0 specifies output port 0 will be enabled

00 specifies no ports enabled
01 specifies port 0 enabled
10 specifies port 1 enabled
11 specifies both ports enabled

Output port availability:

p \[p^{[10]}\] two bit code specifying output ports available
p1 specifies that output port 1 is available
p0 specifies that output port 0 is available

00 specifies no ports are available
01 specifies only port 0 is available
10 specifies only port 1 is available
11 specifies both ports are available

Last used:

u \[u^{[10]}\] two bit code specifying most recently used switch ports
u1=0 specifies port a was most recently used single input
u1=1 specifies port b was most recently used single input
u0=0 specifies port 0 was most recently used single output
u0=1 specifies port 1 was most recently used single output
### A.2. Routing policy control tables

<table>
<thead>
<tr>
<th>rb</th>
<th>ra</th>
<th>p</th>
<th>u</th>
<th>eb</th>
<th>ea</th>
<th>u</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N10</td>
<td>N10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>no outputs are available</td>
</tr>
<tr>
<td>XXX</td>
<td>XXX</td>
<td>00</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>enable no outputs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>neither input has requests</td>
</tr>
<tr>
<td>OXX</td>
<td>OXX</td>
<td>XX</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>enable no outputs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>input b has no requests</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OXX</td>
<td>100</td>
<td>01</td>
<td>XX</td>
<td>00</td>
<td>01</td>
<td>--</td>
<td>input a needs either output</td>
</tr>
<tr>
<td>OXX</td>
<td>100</td>
<td>10</td>
<td>XX</td>
<td>00</td>
<td>10</td>
<td>--</td>
<td>only 0 available: enable 0</td>
</tr>
<tr>
<td>OXX</td>
<td>100</td>
<td>11</td>
<td>X0</td>
<td>00</td>
<td>10</td>
<td>-1</td>
<td>only 1 available: enable 1</td>
</tr>
<tr>
<td>OXX</td>
<td>100</td>
<td>11</td>
<td>X1</td>
<td>00</td>
<td>01</td>
<td>-0</td>
<td>both available: enable u0=1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>both available: enable u0=0</td>
</tr>
<tr>
<td>OXX</td>
<td>101</td>
<td>X0</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>input a needs output 0</td>
</tr>
<tr>
<td>OXX</td>
<td>101</td>
<td>X0</td>
<td>XX</td>
<td>00</td>
<td>01</td>
<td>--</td>
<td>unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>available: enable 0</td>
</tr>
<tr>
<td>OXX</td>
<td>110</td>
<td>0X</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>input a needs output 1</td>
</tr>
<tr>
<td>OXX</td>
<td>110</td>
<td>1X</td>
<td>XX</td>
<td>00</td>
<td>10</td>
<td>--</td>
<td>unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>available: enable 1</td>
</tr>
<tr>
<td>OXX</td>
<td>111</td>
<td>0X</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>input a needs both outputs</td>
</tr>
<tr>
<td>OXX</td>
<td>111</td>
<td>X0</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>1 unavailable</td>
</tr>
<tr>
<td>OXX</td>
<td>111</td>
<td>1X</td>
<td>XX</td>
<td>00</td>
<td>11</td>
<td>--</td>
<td>0 unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>both available: enable both</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>input a has no requests</td>
</tr>
<tr>
<td>100</td>
<td>OXX</td>
<td>01</td>
<td>XX</td>
<td>01</td>
<td>00</td>
<td>--</td>
<td>input b needs either output</td>
</tr>
<tr>
<td>100</td>
<td>OXX</td>
<td>10</td>
<td>XX</td>
<td>10</td>
<td>00</td>
<td>--</td>
<td>only 0 available: enable 0</td>
</tr>
<tr>
<td>100</td>
<td>OXX</td>
<td>11</td>
<td>X0</td>
<td>10</td>
<td>00</td>
<td>-1</td>
<td>only 1 available: enable 1</td>
</tr>
<tr>
<td>100</td>
<td>OXX</td>
<td>11</td>
<td>X1</td>
<td>01</td>
<td>00</td>
<td>-0</td>
<td>both available: enable u0=1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>both available: enable u0=0</td>
</tr>
<tr>
<td>101</td>
<td>OXX</td>
<td>X0</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>input b needs output 0</td>
</tr>
<tr>
<td>101</td>
<td>OXX</td>
<td>X1</td>
<td>XX</td>
<td>01</td>
<td>00</td>
<td>--</td>
<td>unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>available: enable 0</td>
</tr>
<tr>
<td>110</td>
<td>OXX</td>
<td>0X</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>input b needs output 1</td>
</tr>
<tr>
<td>110</td>
<td>OXX</td>
<td>1X</td>
<td>XX</td>
<td>10</td>
<td>00</td>
<td>--</td>
<td>unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>available: enable 1</td>
</tr>
<tr>
<td>111</td>
<td>OXX</td>
<td>0X</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>input b needs both outputs</td>
</tr>
<tr>
<td>111</td>
<td>OXX</td>
<td>X0</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>1 unavailable</td>
</tr>
<tr>
<td>111</td>
<td>OXX</td>
<td>1X</td>
<td>XX</td>
<td>11</td>
<td>00</td>
<td>--</td>
<td>0 unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>both available: enable both</td>
</tr>
<tr>
<td>rb</td>
<td>ra</td>
<td>p</td>
<td>u</td>
<td>eb</td>
<td>ea</td>
<td>u</td>
<td>DESCRIPTION</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>---</td>
<td>---</td>
<td>----</td>
<td>----</td>
<td>---</td>
<td>-------------</td>
</tr>
<tr>
<td>N10</td>
<td>N10</td>
<td>10</td>
<td>ID</td>
<td>10</td>
<td>10</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>110</td>
<td>01</td>
<td>XX</td>
<td>01</td>
<td>00</td>
<td>--</td>
<td>both inputs need different specific outputs</td>
</tr>
<tr>
<td>101</td>
<td>110</td>
<td>10</td>
<td>XX</td>
<td>00</td>
<td>10</td>
<td>--</td>
<td>input a needs output 1, b needs output 0</td>
</tr>
<tr>
<td>101</td>
<td>110</td>
<td>11</td>
<td>XX</td>
<td>01</td>
<td>10</td>
<td>--</td>
<td>only 0 available: enable 0 for b</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>only 1 available: enable 1 for a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>both available: enable both</td>
</tr>
<tr>
<td>110</td>
<td>101</td>
<td>01</td>
<td>XX</td>
<td>00</td>
<td>01</td>
<td>--</td>
<td>input a needs output 0, b needs output 1</td>
</tr>
<tr>
<td>110</td>
<td>101</td>
<td>10</td>
<td>XX</td>
<td>10</td>
<td>00</td>
<td>--</td>
<td>only 0 available: enable 0 for a</td>
</tr>
<tr>
<td>110</td>
<td>101</td>
<td>11</td>
<td>XX</td>
<td>10</td>
<td>01</td>
<td>--</td>
<td>only 1 available: enable 1 for b</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>both available: enable both</td>
</tr>
<tr>
<td>101</td>
<td>101</td>
<td>X0</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>both inputs need output 0</td>
</tr>
<tr>
<td>101</td>
<td>101</td>
<td>X1</td>
<td>0X</td>
<td>00</td>
<td>01</td>
<td>1-</td>
<td>unavailable</td>
</tr>
<tr>
<td>101</td>
<td>101</td>
<td>X1</td>
<td>1X</td>
<td>01</td>
<td>00</td>
<td>0-</td>
<td>only 0 available, favor ( \overline{uT}=0 ): enable for a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>only 0 available, favor ( \overline{uT}=1 ): enable for b</td>
</tr>
<tr>
<td>110</td>
<td>110</td>
<td>0X</td>
<td>XX</td>
<td>00</td>
<td>00</td>
<td>--</td>
<td>both inputs need output 1</td>
</tr>
<tr>
<td>110</td>
<td>110</td>
<td>1X</td>
<td>0X</td>
<td>00</td>
<td>10</td>
<td>1-</td>
<td>unavailable</td>
</tr>
<tr>
<td>110</td>
<td>110</td>
<td>1X</td>
<td>1X</td>
<td>10</td>
<td>10</td>
<td>0-</td>
<td>only 1 available, favor ( \overline{uT}=1 ): enable for a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>only 1 available, favor ( \overline{uT}=0 ): enable for b</td>
</tr>
<tr>
<td>rb</td>
<td>ra</td>
<td>p</td>
<td>u</td>
<td>eb</td>
<td>ea</td>
<td>u</td>
<td>DESCRIPTION</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>---</td>
<td>---</td>
<td>----</td>
<td>----</td>
<td>---</td>
<td>-------------</td>
</tr>
<tr>
<td>N10</td>
<td>N10</td>
<td>10</td>
<td>IO</td>
<td>10</td>
<td>10</td>
<td>IO</td>
<td></td>
</tr>
</tbody>
</table>

**input b needs either output**

| 100 | 101 | 01 | XX | 00 | 01 | -- | input a needs output 0 |
| 100 | 101 | 10 | XX | 10 | 00 | -- | only 0 available: enable for a |
| 100 | 101 | 11 | XX | 10 | 01 | -- | only 1 available: enable for b |

| 100 | 110 | 01 | XX | 01 | 00 | -- | input a needs output 1 |
| 100 | 110 | 10 | XX | 00 | 10 | -- | only 0 available: enable for b |
| 100 | 110 | 11 | XX | 01 | 10 | -- | only 1 available: enable for a |

**input a needs either output**

| 101 | 100 | 01 | XX | 01 | 00 | -- | input b needs output 0 |
| 101 | 100 | 10 | XX | 00 | 10 | -- | only 0 available: enable for b |
| 101 | 100 | 11 | XX | 01 | 10 | -- | only 1 available: enable for a |

| 110 | 100 | 01 | XX | 00 | 01 | -- | input b needs output 1 |
| 110 | 100 | 10 | XX | 10 | 00 | -- | only 0 available: enable for a |
| 110 | 100 | 11 | XX | 10 | 01 | -- | only 1 available: enable for b |

**both inputs need either output**

<p>| 100 | 100 | 01 | 0X | 00 | 01 | -- | only 0 available, favor ( \overline{u1}=1 ): enable for a |
| 100 | 100 | 01 | 1X | 00 | 01 | 0- | only 0 available, favor ( \overline{u1}=0 ): enable for b |
| 100 | 100 | 10 | 0X | 10 | 00 | 1- | only 1 available, favor ( \overline{u1}=1 ): enable for a |
| 100 | 100 | 10 | 1X | 00 | 10 | 0- | only 1 available, favor ( \overline{u1}=0 ): enable for b |
| 100 | 100 | 11 | X0 | 10 | 01 | 0- | both available |
| 100 | 100 | 11 | X1 | 01 | 10 | 1- | both available |</p>
<table>
<thead>
<tr>
<th>rb ra p u</th>
<th>eb ea u</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 111 01 XX</td>
<td>01 00 --</td>
<td>input a needs both outputs, input b either</td>
</tr>
<tr>
<td>100 111 10 XX</td>
<td>10 00 --</td>
<td>only 0 available: enable for b</td>
</tr>
<tr>
<td>100 111 11 XX</td>
<td>00 11 --</td>
<td>only 1 available: enable for b</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both available: enable both for a</td>
</tr>
<tr>
<td>111 100 01 XX</td>
<td>00 01 --</td>
<td>input b needs both outputs, input a either</td>
</tr>
<tr>
<td>111 100 10 XX</td>
<td>00 10 --</td>
<td>only 0 available: enable for a</td>
</tr>
<tr>
<td>111 100 11 XX</td>
<td>11 00 --</td>
<td>only 1 available: enable for a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both available: enable both for b</td>
</tr>
<tr>
<td>101 111 01 XX</td>
<td>01 00 --</td>
<td>input a needs both, input b needs output 0</td>
</tr>
<tr>
<td>101 111 10 XX</td>
<td>00 00 --</td>
<td>only 0 available: enable for b</td>
</tr>
<tr>
<td>101 111 11 XX</td>
<td>00 11 --</td>
<td>0 unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both available: enable both for a</td>
</tr>
<tr>
<td>110 111 01 XX</td>
<td>00 00 --</td>
<td>input a needs both, input b needs output 1</td>
</tr>
<tr>
<td>110 111 10 XX</td>
<td>10 00 --</td>
<td>only 1 available: enable for b</td>
</tr>
<tr>
<td>110 111 11 XX</td>
<td>00 11 --</td>
<td>both available: enable both for a</td>
</tr>
<tr>
<td>111 101 01 XX</td>
<td>00 01 --</td>
<td>input b needs both, input a needs output 0</td>
</tr>
<tr>
<td>111 101 10 XX</td>
<td>00 00 --</td>
<td>only 0 available, enable for a</td>
</tr>
<tr>
<td>111 101 11 XX</td>
<td>11 00 --</td>
<td>0 unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both available: enable for b</td>
</tr>
<tr>
<td>111 110 01 XX</td>
<td>00 00 --</td>
<td>input b needs both, input a needs output 1</td>
</tr>
<tr>
<td>111 110 10 XX</td>
<td>00 10 --</td>
<td>only 1 available, enable for a</td>
</tr>
<tr>
<td>111 110 11 XX</td>
<td>11 00 --</td>
<td>both available: enable for b</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both inputs need both outputs</td>
</tr>
<tr>
<td>111 111 01 XX</td>
<td>00 00 --</td>
<td>only 0 available</td>
</tr>
<tr>
<td>111 111 10 XX</td>
<td>00 00 --</td>
<td>only 1 available</td>
</tr>
<tr>
<td>111 111 10 0X</td>
<td>11 00 1-</td>
<td>both available, favor uX=1: enable both</td>
</tr>
<tr>
<td>111 111 11 1X</td>
<td>00 11 0-</td>
<td>both available, favor uX=0: enable both</td>
</tr>
</tbody>
</table>
B. Input Control Circuit PLA State Table

This section documents the ICC PLA state table in detail. First, all of the required inputs, outputs, and internal state variables are described. Then a set of tables describing the state implementation is given. A general description of the state table is given in §3.

The tables are used as input to the MPLA program to generate the PLA, which is used in the input control circuit ICC for overall control of data flow.
B.1. State variable definition

Input to the PLA:

- en: Output ports available for current request vector
- dp2: Incoming packet is requesting an output port
- dp1: Packet in BSR1 requesting an output port
- dp0: Packet in BSR0 requesting an output port
- fb: Which packet came in first

Output from the PLA:

- s1: Select signal for BSR1
- s0: Select signal for BSR0
- bsel[10]: Select multiplexer for data path through buffers or cut-through
- nbf: Which packet came in first update
- oe2: Enable incoming request vector
- oe1: Enable BCR1 as request vector
- oe0: Enable BCR0 as request vector
- ug: Upstream grant signal

B.2. PLA State Table

<table>
<thead>
<tr>
<th>en</th>
<th>dp2</th>
<th>dp1</th>
<th>dp0</th>
<th>fb</th>
<th>s1</th>
<th>s1</th>
<th>bsel[10]</th>
<th>nbf</th>
<th>oe2</th>
<th>oe1</th>
<th>oe0</th>
<th>ug</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>0 0</td>
<td>0 0</td>
<td>x</td>
<td>0 0</td>
<td>11</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>0 0</td>
<td>1 0</td>
<td>x</td>
<td>0 0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>0 1</td>
<td>0 0</td>
<td>x</td>
<td>0 0</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>1 1</td>
<td>1 x</td>
<td>0 0</td>
<td>11</td>
<td>1</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>0 0</td>
<td>0 x</td>
<td>0 1</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>0 1</td>
<td>1 x</td>
<td>1 0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>1 0</td>
<td>0 x</td>
<td>0 1</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>1 1</td>
<td>1 x</td>
<td>x</td>
<td>xx</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>0 0</td>
<td>0 x</td>
<td>0 0</td>
<td>11</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>0 1</td>
<td>1 x</td>
<td>0 1</td>
<td>00</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>1 0</td>
<td>0 x</td>
<td>1 0</td>
<td>01</td>
<td>x</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>1 1</td>
<td>0 x</td>
<td>0 1</td>
<td>00</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>0 0</td>
<td>0 x</td>
<td>0 0</td>
<td>10</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>0 1</td>
<td>1 x</td>
<td>0 1</td>
<td>00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 0</td>
<td>0 x</td>
<td>1 0</td>
<td>01</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1</td>
<td>1 x</td>
<td>x</td>
<td>xx</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
C. Pin Assignment and Signal Names

This section documents the pin assignment and signal names in detail.

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>signal name</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>blank</td>
<td>sub</td>
<td>Substrate</td>
</tr>
<tr>
<td>2</td>
<td>input</td>
<td>om0</td>
<td>operational mode bit 0</td>
</tr>
<tr>
<td>3</td>
<td>input</td>
<td>pt</td>
<td>packet time</td>
</tr>
<tr>
<td>4</td>
<td>input</td>
<td>om1</td>
<td>operational mode bit 1</td>
</tr>
<tr>
<td>5</td>
<td>input</td>
<td>srsr</td>
<td>soft reset</td>
</tr>
<tr>
<td>6</td>
<td>input</td>
<td>tld</td>
<td>load serial BCR</td>
</tr>
<tr>
<td>7</td>
<td>output</td>
<td>tso</td>
<td>serial output for BCR</td>
</tr>
<tr>
<td>8</td>
<td>input</td>
<td>tsi</td>
<td>serial input for BCR</td>
</tr>
<tr>
<td>9</td>
<td>output</td>
<td>tso1B</td>
<td>enable ICC(B) data mux bit 1</td>
</tr>
<tr>
<td>10</td>
<td>output</td>
<td>tdb0</td>
<td>test data from ICC(B)</td>
</tr>
<tr>
<td>11</td>
<td>output</td>
<td>tdb1</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>12</td>
<td>output</td>
<td>tso0B</td>
<td>enable ICC(B) data mux bit 0</td>
</tr>
<tr>
<td>13</td>
<td>output</td>
<td>tdb2</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>14</td>
<td>output</td>
<td>tdb3</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>15</td>
<td>input</td>
<td>tmei0</td>
<td>mux enable for output ports bit 0</td>
</tr>
<tr>
<td>16</td>
<td>output</td>
<td>tdb4</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>17</td>
<td>output</td>
<td>tdb5</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>18</td>
<td>input</td>
<td>tmei1</td>
<td>mux enable for output ports bit 1</td>
</tr>
<tr>
<td>19</td>
<td>output</td>
<td>tdb6</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>20</td>
<td>output</td>
<td>tdb7</td>
<td>test data form ICC(B)</td>
</tr>
<tr>
<td>21</td>
<td>output</td>
<td>tbo1B</td>
<td>bsel1 for port B</td>
</tr>
<tr>
<td>22</td>
<td>output</td>
<td>tpo1B</td>
<td>port enable signal bit 1 for port B</td>
</tr>
<tr>
<td>23</td>
<td>output</td>
<td>tpo0B</td>
<td>port enable signal bit 0 for port B</td>
</tr>
<tr>
<td>24</td>
<td>output</td>
<td>ddb0</td>
<td>output data on port B</td>
</tr>
<tr>
<td>25</td>
<td>input</td>
<td>tpi0B</td>
<td>port enable signal bit 0 for port B</td>
</tr>
<tr>
<td>26</td>
<td>input</td>
<td>tpi1B</td>
<td>port enable signal bit 1 for port B</td>
</tr>
<tr>
<td>27</td>
<td>input</td>
<td>tpi0A</td>
<td>port enable signal bit 0 for port A</td>
</tr>
<tr>
<td>28</td>
<td>output</td>
<td>err</td>
<td>parity or RC error</td>
</tr>
<tr>
<td>29</td>
<td>input</td>
<td>tpi1A</td>
<td>port enable signal bit 1 for port A</td>
</tr>
<tr>
<td>30</td>
<td>output</td>
<td>tbo0B</td>
<td>bsel0 for port B</td>
</tr>
<tr>
<td>31</td>
<td>output</td>
<td>ddb1</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>32</td>
<td>output</td>
<td>ddb2</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>33</td>
<td>output</td>
<td>ddb3</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>34</td>
<td>output</td>
<td>ddb4</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>35</td>
<td>output</td>
<td>ddb5</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>36</td>
<td>output</td>
<td>ddb6</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>Pin</td>
<td>I/O</td>
<td>signal name</td>
<td>comment</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>37</td>
<td>output</td>
<td>dB7</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>38</td>
<td>output</td>
<td>dB8</td>
<td>output data on output port 1</td>
</tr>
<tr>
<td>39</td>
<td>vdd</td>
<td>Vdd</td>
<td>power supply voltage</td>
</tr>
<tr>
<td>40</td>
<td>gnd</td>
<td>GND</td>
<td>reference voltage; ground</td>
</tr>
<tr>
<td>41</td>
<td>input</td>
<td>dg1</td>
<td>downstream grant for output port 1</td>
</tr>
<tr>
<td>42</td>
<td>gnd</td>
<td>GND</td>
<td>reference voltage; ground</td>
</tr>
<tr>
<td>43</td>
<td>vdd</td>
<td>Vdd</td>
<td>power supply voltage</td>
</tr>
<tr>
<td>44</td>
<td>input</td>
<td>dg0</td>
<td>downstream grant for output port 0</td>
</tr>
<tr>
<td>45</td>
<td>output</td>
<td>ddA8</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>46</td>
<td>output</td>
<td>ddA7</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>47</td>
<td>output</td>
<td>ddA6</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>48</td>
<td>output</td>
<td>ddA5</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>49</td>
<td>output</td>
<td>ddA4</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>50</td>
<td>output</td>
<td>ddA3</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>51</td>
<td>output</td>
<td>ddA2</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>52</td>
<td>output</td>
<td>ddA1</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>53</td>
<td>output</td>
<td>ddA0</td>
<td>output data on output port 0</td>
</tr>
<tr>
<td>54</td>
<td>output</td>
<td>rrf</td>
<td>rotating routing field</td>
</tr>
<tr>
<td>55</td>
<td>output</td>
<td>wdt</td>
<td>watchdog timer</td>
</tr>
<tr>
<td>56</td>
<td>input</td>
<td>tsten</td>
<td>enable to bypass control signals</td>
</tr>
<tr>
<td>57</td>
<td>output</td>
<td>tpo1A</td>
<td>port enable signal bit 1 for port A</td>
</tr>
<tr>
<td>58</td>
<td>output</td>
<td>tpo0A</td>
<td>port enable signal bit 0 for port A</td>
</tr>
<tr>
<td>59</td>
<td>input</td>
<td>tb1</td>
<td>buffer select bit 1</td>
</tr>
<tr>
<td>60</td>
<td>input</td>
<td>tb0</td>
<td>buffer select bit 0</td>
</tr>
<tr>
<td>61</td>
<td>output</td>
<td>tbo0A</td>
<td>buffer select bit 0</td>
</tr>
<tr>
<td>62</td>
<td>output</td>
<td>tbo1A</td>
<td>buffer select bit 1</td>
</tr>
<tr>
<td>63</td>
<td>output</td>
<td>tso0A</td>
<td>BSR 0 shift signal</td>
</tr>
<tr>
<td>64</td>
<td>output</td>
<td>tDA6</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>65</td>
<td>output</td>
<td>tDA7</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>66</td>
<td>output</td>
<td>tso1A</td>
<td>BSR 1 shift signal</td>
</tr>
<tr>
<td>67</td>
<td>output</td>
<td>tDA4</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>68</td>
<td>output</td>
<td>tDA5</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>69</td>
<td>input</td>
<td>tsi1</td>
<td>BSR 1 shift signal</td>
</tr>
<tr>
<td>70</td>
<td>output</td>
<td>tDA2</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>71</td>
<td>output</td>
<td>tDA3</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>72</td>
<td>input</td>
<td>tsi0</td>
<td>BSR 0 shift signal</td>
</tr>
<tr>
<td>Pin</td>
<td>I/O</td>
<td>signal name</td>
<td>comment</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>73</td>
<td>output</td>
<td>tdA0</td>
<td>test data from ICC(A)</td>
</tr>
<tr>
<td>74</td>
<td>output</td>
<td>tdA1</td>
<td>test data for ICC(A)</td>
</tr>
<tr>
<td>75</td>
<td>input</td>
<td>tEn</td>
<td>enable bypass decoder mux</td>
</tr>
<tr>
<td>76</td>
<td>input</td>
<td>tm0</td>
<td>enable ICC(A) data mux bit 0</td>
</tr>
<tr>
<td>77</td>
<td>input</td>
<td>tm1</td>
<td>enable ICC(A) data mux bit 1</td>
</tr>
<tr>
<td>78</td>
<td>output</td>
<td>uDA0</td>
<td>input data on port A</td>
</tr>
<tr>
<td>79</td>
<td>input</td>
<td>phi3</td>
<td>clock phase 3</td>
</tr>
<tr>
<td>80</td>
<td>input</td>
<td>sn0</td>
<td>stage number bit 0</td>
</tr>
<tr>
<td>81</td>
<td>input</td>
<td>sn1</td>
<td>stage number bit 1</td>
</tr>
<tr>
<td>82</td>
<td>input</td>
<td>sn2</td>
<td>stage number bit 2</td>
</tr>
<tr>
<td>83</td>
<td>input</td>
<td>hrst</td>
<td>hard reset</td>
</tr>
<tr>
<td>84</td>
<td>input</td>
<td>phi1</td>
<td>clock phase 1</td>
</tr>
<tr>
<td>85</td>
<td>input</td>
<td>uDA1</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>86</td>
<td>input</td>
<td>uDA2</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>87</td>
<td>input</td>
<td>uDA3</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>88</td>
<td>input</td>
<td>uDA4</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>89</td>
<td>input</td>
<td>uDA5</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>90</td>
<td>input</td>
<td>uDA6</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>91</td>
<td>input</td>
<td>uDA7</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>92</td>
<td>input</td>
<td>uDA8</td>
<td>input data for input port A</td>
</tr>
<tr>
<td>93</td>
<td>vdd</td>
<td>Vdd</td>
<td>power supply voltage</td>
</tr>
<tr>
<td>94</td>
<td>gnd</td>
<td>GND</td>
<td>reference voltage; ground</td>
</tr>
<tr>
<td>95</td>
<td>output</td>
<td>uGA</td>
<td>upstream grant for input port A</td>
</tr>
<tr>
<td>96</td>
<td>gnd</td>
<td>GND</td>
<td>reference voltage; ground</td>
</tr>
<tr>
<td>97</td>
<td>vdd</td>
<td>Vdd</td>
<td>power supply voltage</td>
</tr>
<tr>
<td>98</td>
<td>output</td>
<td>uGB</td>
<td>upstream grant for input port B</td>
</tr>
<tr>
<td>99</td>
<td>input</td>
<td>uDB8</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>100</td>
<td>input</td>
<td>uDB7</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>101</td>
<td>input</td>
<td>uDB6</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>102</td>
<td>input</td>
<td>uDB5</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>103</td>
<td>input</td>
<td>uDB4</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>104</td>
<td>input</td>
<td>uDB3</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>105</td>
<td>input</td>
<td>uDB2</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>106</td>
<td>input</td>
<td>uDB1</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>107</td>
<td>input</td>
<td>uDB0</td>
<td>input data for input port B</td>
</tr>
<tr>
<td>108</td>
<td>input</td>
<td>phi2</td>
<td>clock phase 2</td>
</tr>
</tbody>
</table>
D. Interconnection of Circuit Blocks

This section shows a detailed diagram of the chip.

Figure 17: Chip block diagram