SHAP3

ATecoM

ATM Traffic Shaper
(Generator/Receiver)
SHAP3

Title:	ATM Traffic Shaper SHAP3
Version:	2.0
Last update:	14. November 1996
Originator:	M. Guth, T. Lux

Purpose:	Specify Shaper implementation
Keywords:	ATM, UPC, Shaping, Measurement

Features

- Shaping on a per class basis
- Supports 32 classes
- Single or Dual Leaky Bucket Shaping function
- Allows control of PCR, SCR, MBS and CDV traffic parameters
- Allows 32 output rates per class per LB
- Select different parameter sets by HW or SW
- Up to 2M cells/class
- Uses low cost 60/70 ns Dynamic RAM
- Utopia level 1 and SAI compatible interfaces
- Output can be halted while arriving cells are stored
- Supports 53 and 56 byte cell size
- Cell size conversion 53 ↔ 56 bytes
- Cell drop and insert via microprocessor interface
- Repeated cell sequence transmission at specified rate
- 24 bit time stamp for cell arrival timing

Draft	Approval	Date	Reason for change
1.0		31.08.1995	Design Tape-out
1.1		08.03.1996	Testing of device
2.0		14.11.1996	

Table of Contents

Table of Figures

1 General description

The idea of traffic shaping in ATM is well known. However there are numerous ways to perform traffic shaping. This document describes how traffic shaping is done by the ATecoM SHAP3 device.

The following figure depicts the block diagram of the SHAP3:

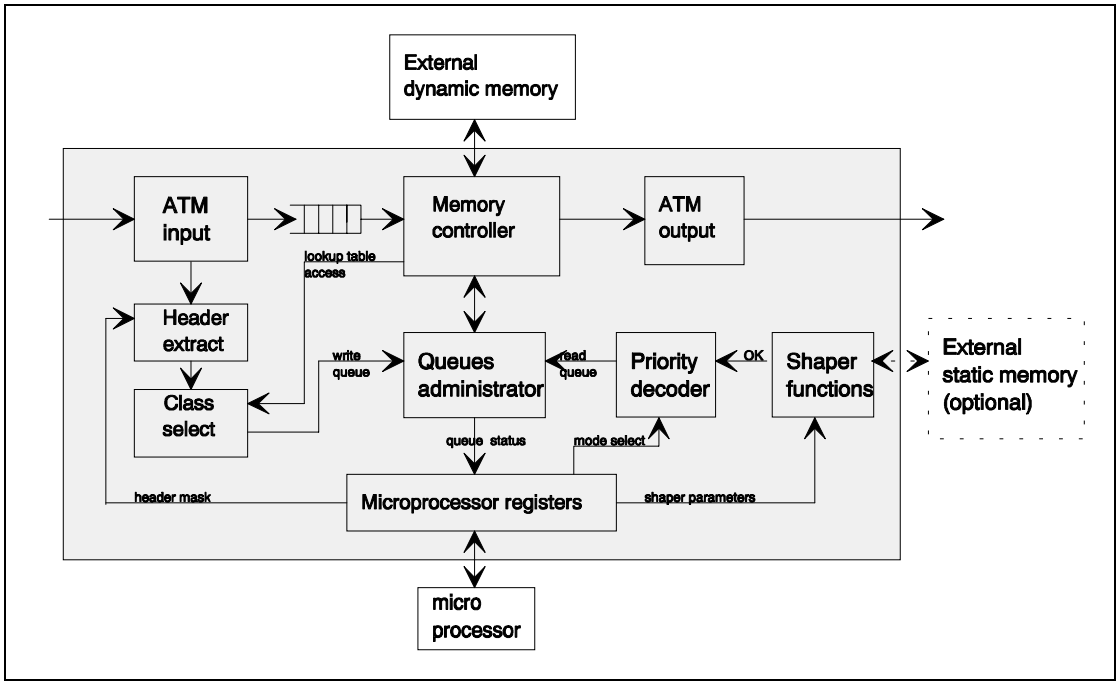


Figure 1-1: Block diagram of the SHAP3

The SHAP3 is a device which allows an incoming ATM cell stream to be output with a pre-defined rate. To do so the cell stream is split into 32 traffic classes. For each class the following parameters can be set:

- Peak Cell Rate output (PCR)
- Sustainable Cell Rate output (SCR)
- Maximum Burst size at PCR output (MBS)
- CLP bit handling
- Amount of memory used by the class
- Priority within the classes

The output rate is controlled according to one of two Shaper functions both based on the Leaky Bucket algorithm namely the Single Leaky Bucket and the Dual Leaky Bucket.

The SHAP3 operates using a number of classes. Up to 16 bits of the arriving cell header are extracted and presented to a lookup table in the main memory. From the table is extracted the class to which the cell belongs. The cells of each class are stored in a queue which is also held in the external memory.

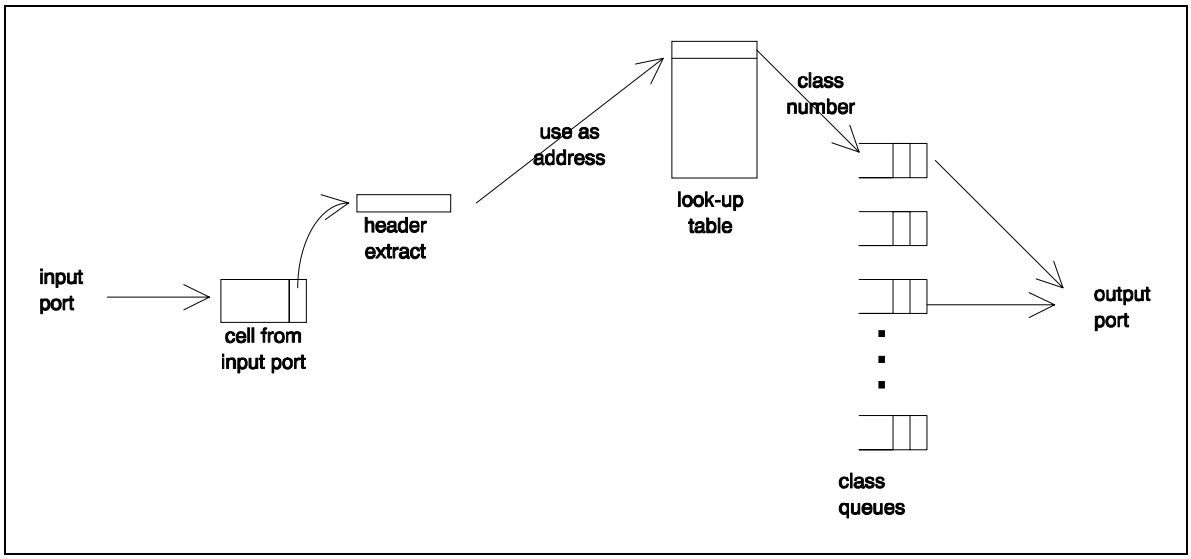


Figure 1-2: Class selection

Arriving cells are stored until the queue is full. If a queue is full and more cells arrive they are discarded. A counter keeps track of the number of discarded cells. Each class has a Shaper function assigned to it. The Shaper function determines when a cell is allowed to leave the queue. However the system has several queues thus it can happen that multiple Shaper functions signal that a cell should be output. If multiple request to output a cell are made, a priority function decides which request is honoured. The device supports static priorities as well as round robin priorities.

To support flow control the device allows the rate at which data is output to be changed under hardware or software control (see chapter 8 Class Rate Adaptation).

Beside shaping an ATM stream passing through the device, it is also possible to start or terminate a stream at the device. This allows the SHAP3 to be used as Traffic enerator, Traffic analyser or low speed AAL0.

2 ATM ports

The SHAP3 support the SAI- and the Utopia interface. The input port is decoupled from the rest of the chip through a four cell FIFO. Cells will first be completely stored in the FIFO before they are handled by the device. The output of the SHAP3 can be stopped in which case new arriving cells are stored in the queues. The SHAP3 can work with a 53 or 56 byte cell stream. In the 56 byte mode the first 3 bytes may be used for a routing tag. It is also possible to convert 56 byte cells to 53 byte cells.

2.1 Input port

The input port of the SHAP3 has the following properties:

- can work on a continuous or discontinuous cell stream
- can be enabled or disabled
- can check or ignore an incoming HEC
- can pass or remove incoming idle cells
- can work on 53 or 56 byte cells
- can insert a three byte time stamp at the end of the cell when running in 53 byte cell mode
- can convert from 56 to 53 byte cell format

Bit 0 of the *IOC register* enables or disables the input port.

- If bit 0 of the *IOC register* is cleared all incoming cells are discarded.
- If bit 0 of the *IOC register* is set the input port is enabled.

The input port of the device is decoupled through a four cell FIFO. The FIFO is used for clock de-coupling only and not for rate adaptation.

2.1.1 Cell alignment

In ATM two different cell streams are possible:

- systems where a continuous cell stream is present and all space between cells containing valid data is filled with idle cells.
- systems with a discontinuous cell stream where the cell stream will contains gaps of arbitrary length.

The input port works correctly on both types of data streams. In all cases the start of a cell is marked with the so called cell sync signal which is active during the arrival of the first byte of a cell. As soon as the first byte has entered the device the following 52 (or 55) bytes are assumed to belong to the same cell and are stored accordingly. If during the loading of these 52 (or 55) bytes another cell sync arrives this signal is ignored. The event is registered in the *input status register* and can provide an interrupt, signalling that an early sync has been detected.

The input port can also notice if there is no cell sync immediately following the last loaded cell byte: Late Sync Detection. This is an error when a continuous cell stream should be present. This event can also be registered in the input status register and can also provide an interrupt signal. Systems working with a discontinuous cell stream should simply ignore this status bit.

If the RxEnb* signal is deasserted this indicates to the device sending cells that it should stop sending data. If the input stream stops there will be no cell sync immediately following the last loaded cell byte. In that case the Late Sync Detection bit is also set.

2.1.2 Utopia input port signals

The Utopia interface is meant to couple devices together even if the devices have different operating speeds. The SHAP3 has the task to adapt the cell stream between two devices operating on different speeds. The input port of the SHAP3, although it can be connected to a Utopia port, behaves somewhat different to a normal Utopia interface. The system clock of the SHAP3 must be high enough for the device to receive the maximum data rate and store the incoming cells in the memory. The output rate is then determined by the dual Leaky Bucket shaper function.

At the input port there is no input signal for the Utopia CLAV signal. This is not needed as the SHAP3 accepts data at any time. The SHAP3 can receive cells as long as there is room in the input FIFO.

If the RxEnb* signal is deasserted and the user ignores this signal and keeps transmitting cells towards the SHAP3 the cells are still processed. That is the SHAP3 will correctly receive the cells and store them in a queue. Cells are lost on two conditions:

1. If the input FIFO overflows
2. If the queue to store the arriving cell is full

The four cell input FIFO gets full

If the input FIFO gets full the SHAP3 can be programmed to deasserted the RxEnb* signal. However this can happen only if the device can not read the cells fast enough from the FIFO. This is normally a system design error as the system clock must be fast enough for the SHAP3 to read the cells from the FIFO and write the cells in the main memory.

The queue is full

This is an indication that the traffic is too bursty or the queue is too small. However in this case an error counter keeps track of the number of lost cells.

The RxEnb* signal will be deasserted under two conditions:

1. If the TxClav is deasserted.
2. If the input FIFO gets full.

The RxEnb* signal can be programmed for four different modes:

- If bit 3 of the *IOC register* is cleared the RxEnb* signal is asserted at the end of a cell only.
- If bit 3 of the *IOC register* is set the RxEnb* signal is asserted directly.
- If bit 7 of the *IOC register* is cleared the RxEnb* signal is asserted when the TxClav signal is deasserted.
- If bit 7 of the *IOC register* is set the RxEnb* signal is asserted if the TxClav signal is deasserted and the input FIFO is full.

2.1.3 Idle cells

The input port can remove incoming idle cells or pass them on. If idle cell detection is switched on, the device checks whether an incoming cell has an idle cell header (first four bytes) or not. Idle cells are discarded and are not passed through the device. If idle cell detection is switched off all cells are handled by the device.

Bit 2 of the *IOC register* controls the storage of idle cells.

- If bit 2 of the *IOC register* is cleared idle cells are treated as normal data cells.
- If bit 2 of the *IOC register* is set idle cells are removed from the data stream.

2.1.4 HEC check

The input port can check the cell HEC field and discard the cell if an HEC error occurs. It is not possible to correct the HEC field. This HEC check option can be disabled, allowing the HEC position to contain arbitrary data.

Bit 1 of the *IOC register* controls the check of the cell header.

- If bit 1 of the *IOC register* is cleared the HEC byte is ignored.
- If bit 1 of the *IOC register* is set the HEC is checked against the HEC-byte. If a HEC error occurs the cell is discarded.

2.1.5 Cell format

The input controller supports three different cell formats:

- 53 byte cells
- 53 byte cells with 3 leading tag bytes
- 56 byte cells

It can also perform cell format conversion from 56 to 53 bytes. In this mode the first three octets of each cell are removed and the system clock must fulfil the following condition:

$$\text{systemclk} \geq \frac{53\ \text{bytes}}{56\ \text{bytes}} \bullet \text{RxClk}$$

Bits 5 and 6 of the *IOC register* control the type of cells coming into the device:

IOC [6:5]	Input cell type	Write to RAM
00	53 bytes	53 byte + 24 bit time stamp
01	53 bytes + 3 tag bytes	53 byte + 24 bit time stamp
10	56 bytes	56 bytes
11	53 bytes + 3 tag bytes	53 bytes + 3 tag bytes

The three different cell formats are shown below:

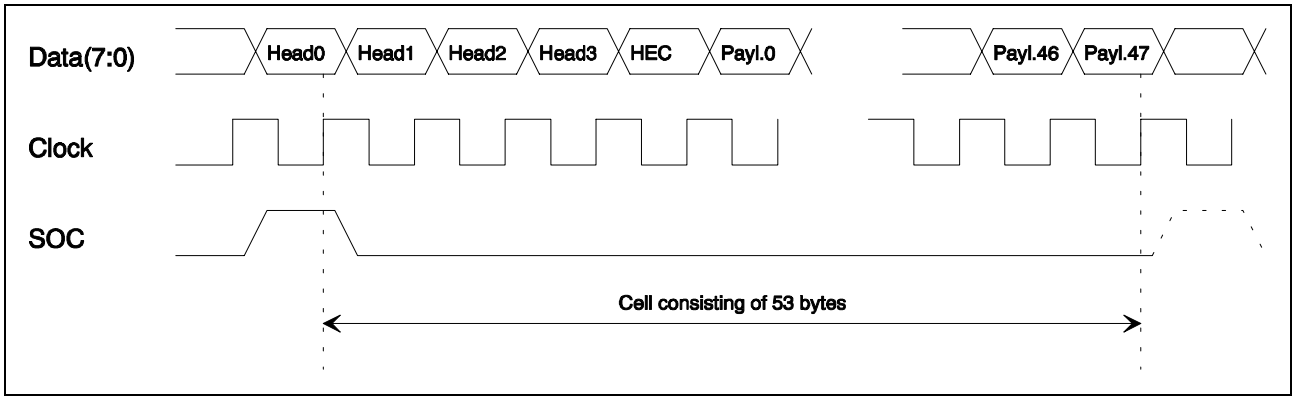


Figure 2-1: 53 Byte cell format

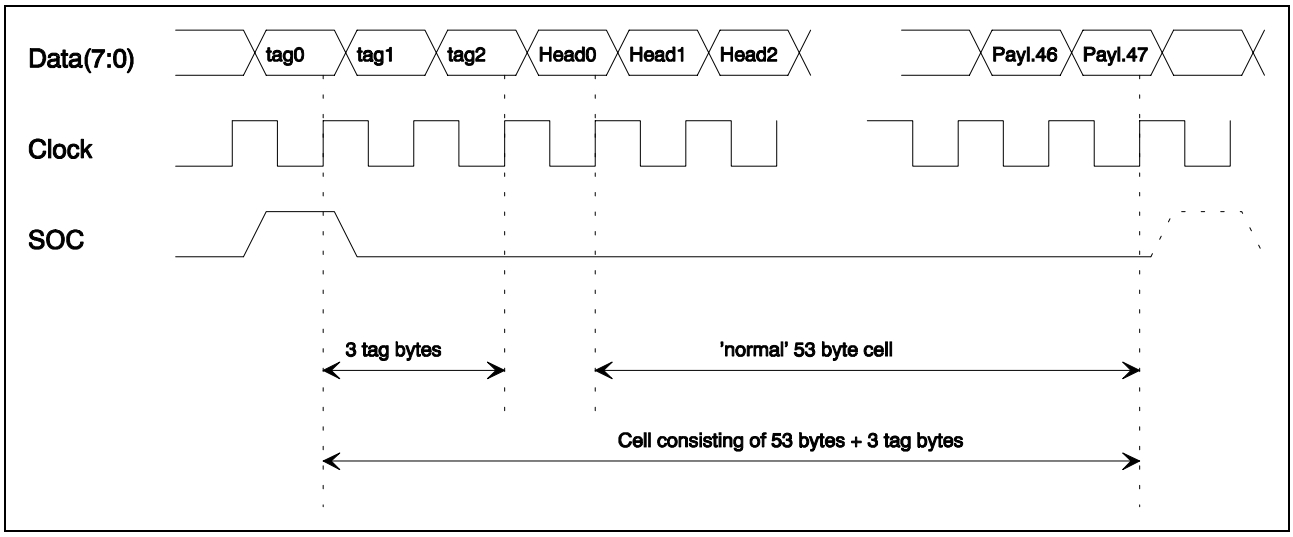


Figure 2-2: 53 Byte + 3 routing tag bytes cell format

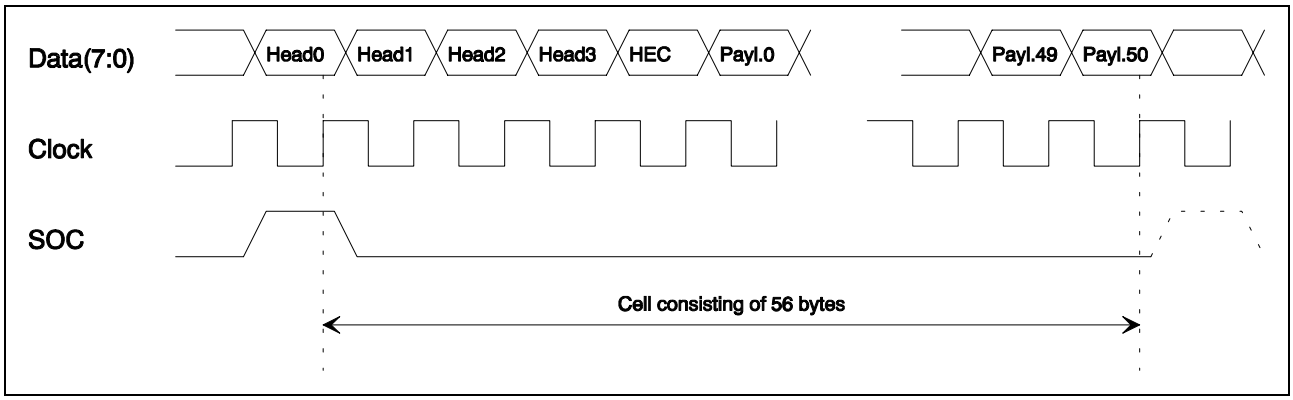


Figure 2-3: 56 Byte cell format

The difference between a 56-byte cell and a 53+3 tag bytes cell is the location of the header in the cell stream. This is important as the SHAP3 uses the information in the header to assign cells to a certain queue.

2.1.6 Time stamp

As described above the controller can write a 24 bit time stamp indicating the arrival time of the cell in the memory. For this time stamp two different resolutions are available:

- If bit 4 of the *IOC register* is cleared the time stamp counter is incremented each RxClk.
- If bit 4 of the *IOC register* is set the time stamp counter is incremented once every 53 or every 56 RxClk cycles depending on the selected input cell format.

An increment every RxClk provides the best accuracy. However the time stamp counter will roll over after some time. Before this happens a cell must have arrived. Thus the minimum distance between cells must be 2^{24} clock cycles. This is about one cell every second.

An increment every cell time allows a longer period of non-arriving cells: 53 or 56 seconds.

2.1.7 Input status

The device has a register that reflects the status of the input port. The following events can be noticed:

- Early cell sync
- Late cell sync
- Missing RxClk
- Discarded cell counter rollover

- FIFO level high
- FIFO overflow

Early cell sync and late cell sync have been explained in chapter 2.1.1 Cell alignment.

Missing RxClk

The incoming RxClk is checked against the system clock. If the incoming RxClk is not present for 32 system clock cycles the missing RxClk bit will be set.

Discarded cell counter overflow

Each queue has a counter counting the number of discarded cells. This counter can be programmed to stay at it's maximum value or roll over and set a status bit. There is only a single status bit for all counters.

FIFO level high

A FIFO level high occurs when more than two cells are in the input FIFO. The reason may be that the input cellrate is higher than the output cellrate.

FIFO overflow

A FIFO overflow occurs when the input data rate is higher than the output data rate. The device will discard all arriving cells when the FIFO is full and sets the FIFO full bit. The device has a four cell FIFO which is under normal conditions large enough to handle clock variations and flow control between input and output. Although the device is not meant for rate decoupling[1] it can be used as such under certain conditions. It can always perform rate decoupling from a low rate towards a high rate. If the input port removes incoming idle cells and the rate of incoming data cells is sufficient low some rate decoupling from high to low rate can be performed.

Cells discarded due to a FIFO overflow are not counted in the class discarded cell counters.

Clearing of status bits

The input status bits will remain set until cleared by a write from the microprocessor. Clearing a status bit is done by writing a "1" to the corresponding position in the *input status register*. This "1" is not stored. It is only used to clear the status bit. Each *input status register* has a related *interrupt control register*. When a bit in the *interrupt control register* is set the corresponding bit in the *input status register* will cause an interrupt to occur.

2.2 Output port

The output port is 8 bits wide. As with the input port, the output port can work in continuous or discontinuous mode. The output port has no separate cell-clock signal, instead the system clock can be used as the cell-clock signal.

2.2.1 Utopia output port signals

The TxSOC signal is active during the first octet of a cell is present a the ATM output port. As long as the TxData bus holds valid data, the TxEnb* signal is asserted (set to low).

The SHAP3 works on cell-slot basis. This means that the SHAP3 aligns all cells on a 53 (or 56) clock cycle intervals. In an interval either a valid cell will leave the SHAP3 or not. This also implies that if the output should stop sending data it can not be stopped for an arbitrary time interval but only for a complete cell time. The point at which the TxClav signal is sampled is during the rising clock edge of byte 31 of the cell. This is equal for a 53 and a 56 byte cell. Stopping the output means that the buckets inside the SHAP3 will continue to leak. Thus it can happen that after the output is enabled again the SHAP3 will produce a burst of cells.

[1]For real rate decoupling much larger FIFOs are needed.

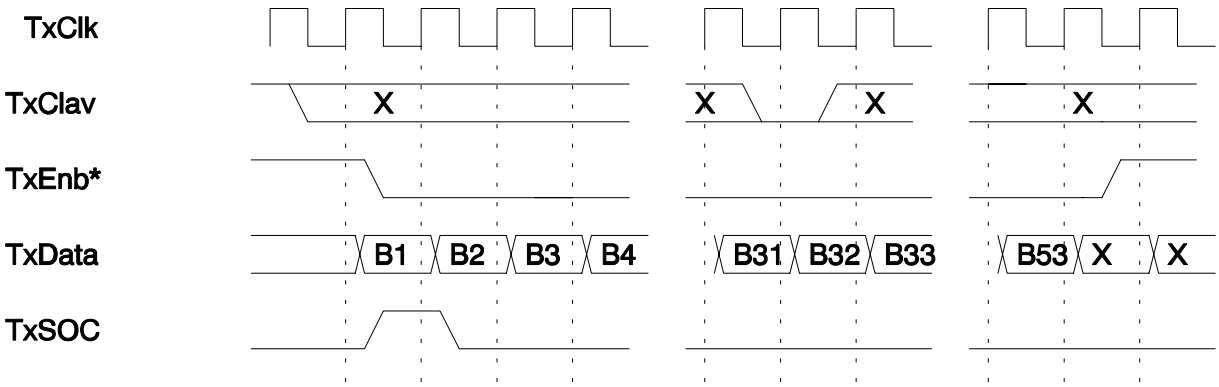


Figure 2-4: Stopping cell output for 53 byte cells

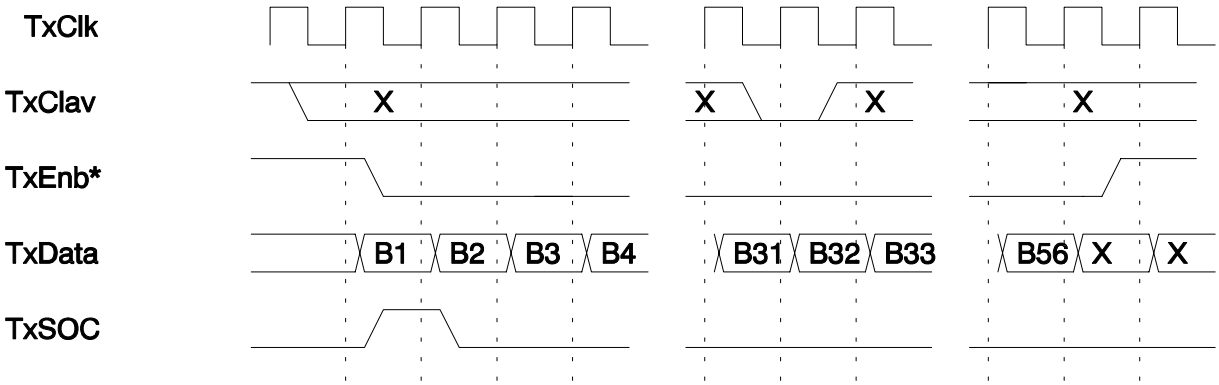


Figure 2-5: Stopping cell output for 56 Byte cells

2.2.2 Idle cell production

In continuous mode the output generates idle cells if no cell is allowed to leave the SHAP3 or all the queues are empty. In discontinuous mode the device outputs no data if no cell is available. In that case there will be gaps between the cells. The gaps will always be 53 (or 56) bytes long as the SHAP3 works on cell slot basis. This means that the SHAP3 aligns all cells on 53 (or 56) system clock cycle intervals. In an interval either a cell will leave the SHAP3 or not.

Bit 0 of the *COLB register* controls the generation of idle cells.

- If bit 0 of the *COLB register* is cleared idle cells are not produced at the output and there will be gaps between cells if no data cell is output.
- If bit 0 of the *COLB register* is set an idle cell is generated at the output port if no data cell is output.

2.2.3 Output cell format

The *output control register* determines the output cell format of the SHAP3.
Bit 1 of the *COLB register* controls the type of cells leaving the device:

- If bit 1 of the *COLB register* is cleared cells of 53 bytes leave the SHAP3.
- If bit 1 of the *COLB register* is set cells of 56 bytes leave the SHAP3.

This bit also influences the rate of the SHAP3 related to the system clock. The Leaky Bucket calculations are performed once per cell time. For an outgoing cell format of 53 bytes a cell time equals 53 system clock cycles. For an outgoing cell format of 56 bytes a cell time equals 56 system clock cycles. This bit should only be changed during initialisation of the SHAP3. **It is not allowed to change this bit while the cell stream is running**.

The cell type of the *IOC register* specifies the cell format stored in the memory. Bit 1 of the *COLB register* specifies the format of the outgoing cell. These two functions are completely independent although some combinations give strange results like:

- Input cell size is 56 bytes but output format is 53 bytes.
  In this case the last 3 bytes of a cell are lost.

- Input format is 53 bytes, output format is 56 bytes.
  The cell will contain 3 extra bytes at the end that hold the 24 bit time stamp.
  (Note that to operate in this mode the following condition must be fulfilled: $\text{systemclk} \geq \frac{56\ \text{bytes}}{53\ \text{bytes}} \bullet \text{RxCLk}$)

As the SHAP3 can work with two different cell sizes throughout this document the description ***53 (or 56)*** is found. If the cell size programmed in the ATM *output control register* is 53 the system works in a cadence of 53 clock cycles. (Thus storing and reading a cell takes 53 clock cycles, a refresh is performed every nx53 clock cycles, a class increment or decrement can be performed once every 53 clock cycles, etc.). If the programmed cell size in the ATM *output control register* is 56 the system works in a cadence of 56 clock cycles.

Note that the input stage of the SHAP3 works on the RxClk and thus runs independent from the rest of the chip.

2.2.4 Idle cell format

As the 56 byte cell is non-standard there is no idle cell format defined for such cell sizes. The SHAP3 provides two different cell formats for idle cells:

- If bit 2 of the *COLB register* is cleared an idle cell consists of 3 routing tag bytes all zero followed by a normal 53 byte idle cell pattern.
- If bit 2 of the *COLB register* is set an idle cell starts as a normal idle cell but with 51 bytes of idle cell payload.

If the cell size is 53 bytes (bit 1 of *COLB register* is cleared) bit 2 is don't care. Otherwise bit 2 specifies the 56 byte idle cell format.

2.2.5 Circular mode

Circular mode is provided to use the SHAP3 as a continuous traffic generator device. The device supports beside the normal operation mode three circular modes. In normal mode cells are read from a queue as long as the queue holds data. This guarantees that the SHAP3 will not output any cells if the queue is empty.

In circular mode this protection is disabled for some, or all the queues. As a consequence the SHAP3 will output all cells from a queue and then start reading at the beginning again. In this mode the SHAP3 will transmit the complete queue contents over and over again. No other aspect of the SHAP3 is changed. Those queues which are not in circular mode function normally.

Bits 5 and 6 of the *COLB register* control the circular mode.

3 Shaping function

The shaper function determines the characteristic of the outgoing data stream. Two operation modes are available: Dual Leaky Bucket or single Leaky Bucket mode. The Leaky Bucket mode is determined by bit 7 of the *Leaky Bucket control register*. In dual Leaky Bucket mode each class has one dual Leaky Bucket parameter (if the external SRAM is used up to 32 dual Leaky Bucket parameter per class). In single Leaky Bucket mode each class has two single Leaky Bucket parameters (if the external SRAM is used up to 64 single Leaky Bucket parameter per class). This mode allows output control with two different single Leaky Bucket parameters without the need of the external parameter RAM.

In single Leaky Bucket the output rate is controlled with a single Leaky Bucket. This allows a user to set the Peak Cell Rate (PCR) and the Cell Delay Variation (CDV) of the outgoing cell stream.

In dual Leaky Bucket mode the output rate is controlled with two Leaky Buckets in parallel. A cell is output if both Leaky Buckets give their OK. This allows cells to be output at a specified Peak Cell rate, Sustainable Cell Rate, Maximum Burst Size at PCR and CDV. This gives a three stage rate control.

3.1 Rate parameters

The output rate is based on the system clock together with the cell size. The Leaky Bucket calculations are performed once per cell time. For a 53 byte outgoing cell size a cell time equals 53 system clock cycles. For a 56 byte outgoing cell size a cell time equals 56 system clock cycles.

Example 1:

- System clock = 19.44 MHz
- Cell size = 53 bytes

Systemrate is 19.44 10^6 Hz/53 bytes = 366792 cells/sec (=155.52 Mbit/sec)

Example 2:

- System clock = 23 MHz
- Cell size = 53 bytes

Systemrate is 23 10^6 Hz/53 bytes = 433962 cells /sec (=184 Mbit/sec)

Based on that, the rate is further determined by the following parameters:

Limit:

The bucket limit together with the splash determines the maximum burst size that can be output (leak<<splash). The bucket limit value is 24 bits wide. As long as the level is below or equal to the limit a cell is allowed to leave the queue.

Splash:

This value together with the leak determines the output rate. The splash value is 8 bits wide. The splash is the amount added to the level for each cell that is leaving the queue. The splash is only added if a cell is actually leaving the queue. Not if a cell-request is made.

Leak:

The leak is the amount with which the level is decreased per cell time. The leak parameter is often taken to be 1 and therefore not explicitly mentioned in numerous descriptions of the Leaky Bucket algorithm. This value together with the splash determines the output rate. The leak value is stored in a coded 4 Bit format The table below shows how the leak rate is coded.

Code	Leak rate	Code	Leak rate
0x0	128	0x8	½
0x1	64	0x9	¼
0x2	32	0xA	1/8
0x3	16	0xB	1/16
0x4	8	0xC	1/32
0x5	4	0xD	1/64
0x6	2	0xE	1/128
0x7	1	0xF	0

Level:

The level is changing depending on the cell flow and the other three parameters Limit, Splash and Leak. This value is 31 bits wide where 7 bits are fraction bits (because the minimum Leak is 1/128). The Level can be pre-set by the user when setting up a connection. This can be used to avoid that the first cells of this connection are output back to back (if CDV is allowed).

3.2 Special rates

3.2.1 Full data stream

The SHAP3 can be programmed to pass all cells of a dedicated class. No shaping is performed. The Leaky Bucket parameter for that class would be:

Bucket limit = 0xFFFFFF

Splash = 0x0

Leak code = 0xF (no leaking)

Level = don't care

So the Level is always lower than or equal to the Limit. In dual Leaky Bucket mode both parameters should be programmed this way.

3.2.2 Stop data stream

The SHAP3 can be programmed to stop the data stream of a dedicated class. The Leaky Bucket parameter would be:

Bucket limit **= 0x0**

Splash **= 0x0**

Leak code **= 0xF (no leaking)**

Level **= 0xFFFFFF**

So the Level is always higher than the Limit. In dual Leaky Bucket mode only one parameter need to be programmed this way

3.3 Single Leaky Bucket

3.3.1 Rate control

The Peak Cell Rate (PCR) is set with the **leak** and **splash** parameters according to:

$$\mathbf{PCR} = \frac{\mathbf{leak}}{\mathbf{splash}} \bullet \mathbf{systemrate}$$

The splash is normally chosen in the range 128..255 because this gives the best accuracy. A rate change is then possible in steps of $\frac{1}{128}$ (accuracy 0.78%) to $\frac{1}{255}$ (accuracy 0.39%).

The minimum PCR can then be calculated from the minimum leak and maximum splash:

$\frac{1}{128} \bullet \frac{1}{255} = \frac{1}{32640}$.

With a linkrate of 155 Mbits/sec this gives a minimum rate of 4.8 Kbits/sec. It is also possible to select a splash from a different range e.g. 64..127 or 32..63. The accuracy is then 1.56% or 3.12%. Several accuracy values are available between 0.78% and 50%.

For example: With an accuracy ≤ 3.12% we have the following parameters:

- Minimum rate 38 Kbit/sec
- Maximum rate 155 Mbit/sec
- Maximum number of consecutive cells: 256K

The maximum number of consecutive cells is determined by:

$$\mathbf{Maximum\ number\ of\ consecutive\ cells} = \left\lfloor \frac{\mathbf{limit}}{\mathbf{splash\ -\ leak}} \right\rfloor + \mathbf{1} \approx \left\lfloor \frac{\mathbf{limit}}{\mathbf{splash}} \right\rfloor + \mathbf{1} \quad (\mathbf{leak} << \mathbf{splash})$$

At all rates a number of 64K consecutive cells can be obtained. If the limit is set to zero the number of consecutive cells is one. In this case the SHAP3 tries to output all cells equidistant (spacer[2]). It is still possible that due to a higher priority class, cells will leave the SHAP3 not equidistant. This is not the case for class zero in absolute priority mode as there is no class with a higher priority.

The CDV generated by the Leaky Bucket is determined by:

$$\mathbf{CDV} = \left\lfloor \frac{\mathbf{limit}}{\mathbf{leak}} \right\rfloor [celltime] \Rightarrow \mathbf{leak} \bullet \mathbf{CDV} \le \mathbf{limit} < (\mathbf{leak} + \mathbf{1}) \bullet \mathbf{CDV}$$

The CDV is defined to be a positive integer in celltime units.

Note that CDV and the maximum number of consecutive can not be set independently.

The minimum CDV is zero (limit = 0). The maximum CDV depends on the leak rate and the limit. From the formula it can be seen that the largest CDV is reached with low leak rates or a high limit.

If the leak rate and the CDV are given the necessary limit can be calculated by the formula:

$$\mathbf{limit} = \lceil \mathbf{leak} - \mathbf{1} \rceil + \lfloor \mathbf{leak} \bullet \mathbf{CDV} \rfloor$$

[2]Note that a cell shaper is more powerful than a spacer.

The single Leaky Bucket controls PCR and CDV. The Leaky Bucket implementation of ATecoM outputs a cell from the SHAP3 if the level is below or equal to the limit. Thus if the limit is set to zero cells are still allowed to leave the SHAP3.

3.3.2 Examples

With a limit set to zero the CDV is zero (See formulas above). Thus all cells will leave the SHAP3 equidistant. Figure 3-1 shows how the level changes as function of the parameters with a limit of zero.

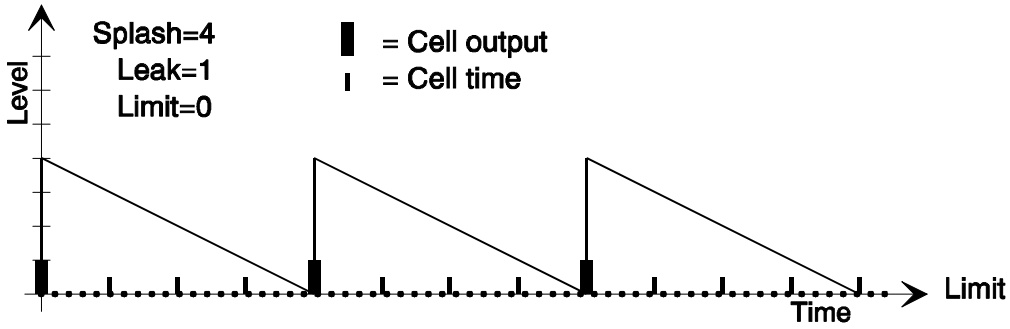


Figure 3-1: Single Leaky Bucket with CDV of zero cell times

With a limit of three we get a CDV of three cell times (two consecutive cells). Figure 3-2 shows this case.

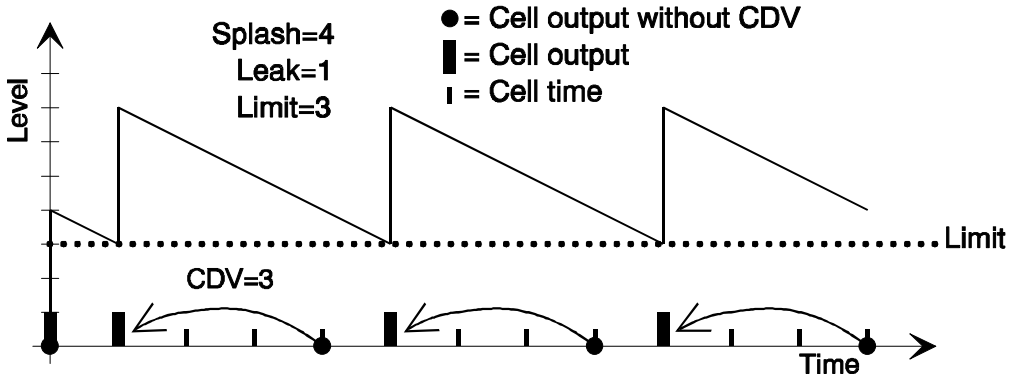


Figure 3-2: Single Leaky Bucket with a CDV of three cell times

If we have Leak=1, Splash=4, Limit=2 we get a CDV of two cell times and a number of consecutive cells of one (Figure 3-3).

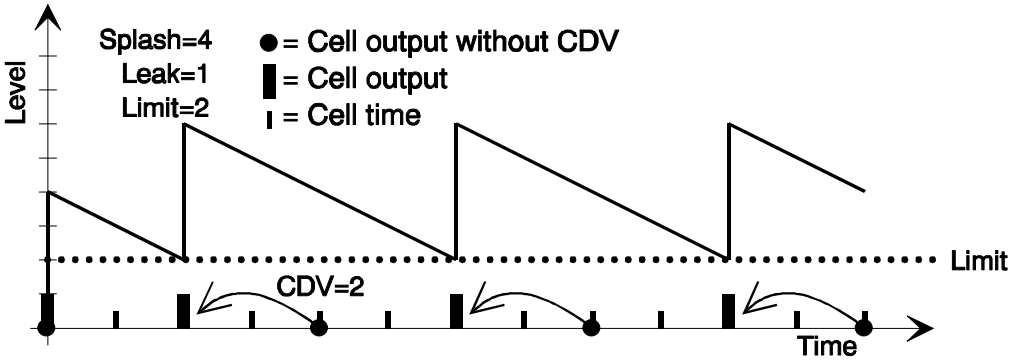


Figure 3-3: Single Leaky Bucket with a CDV of two cell times

Consecutive cells could be sent only if the level is dropped far enough below the limit to allow one or more cells. This happens if no cells are output for some time.

There are three cases when no cell is output although the shaper function requests one:

1. There are no more cells in the queue.

2. Cells are output from another class.
3. The output from the SHAP3 is halted (TxClav signal is low).

The first case happens if the queue is empty, which can only happen if no new cells arrive for the class. Thus an output burst will appear if several cells arrive at the input of the class after a period of rest. The second case can happen at all times except for the class with the highest priority. The third case depends on the hardware configuration in which the device is used. Figure 3-4 shows a typical figure of the SHAP3 output when a large burst of cells arrive. At the moment the burst arrives the bucket level is zero. This is the case if the class has been enabled recently or the queue has been empty for some time.

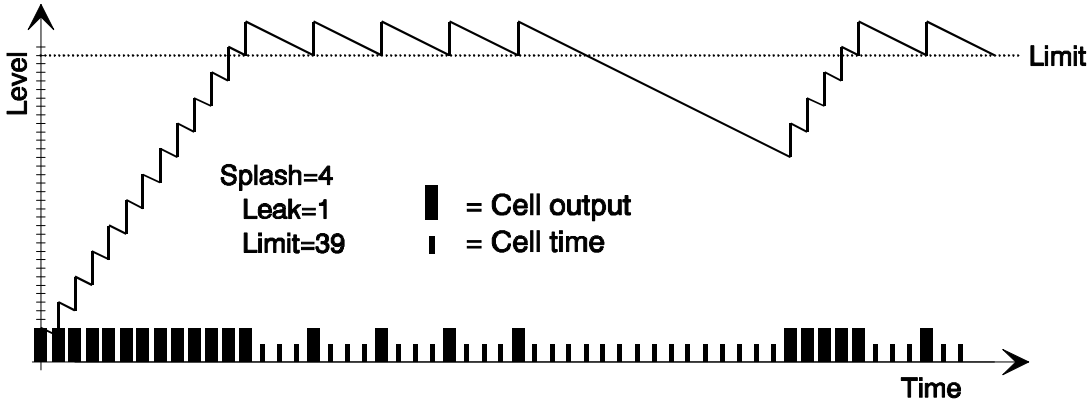


Figure 3-4: Typical level behaviour

The level is far below the limit thus a large bursts of cells is output while the level rises to the limit. When the limit has been reached the PCR of 1:4 is active unless one of the three cases described above occurs. If no cells are sent for some period the level drops as the bucket is leaking continuously. When cells can be output again, a burst of cells is sent until the limit has been reached again. The setting of a certain CDV can also be used to output a more regular cell flow when more than one class is active. In chapter 4.2 (Queue output conflicts) an example is given.

In case of large CDV values generated by the single Leaky Bucket algorithm a lot of cells are output consecutively. The disadvantage is that all these cells are output with the full link rate. This is often intolerable.

3.4 Dual Leaky Bucket

With a single Leaky Bucket the Peak Cell Rate (PCR) and the CDV can be controlled. However the Maximum Burst Size (MBS) at PCR and the Sustainable Cell Rate can not be controlled. To control all four parameters a dual Leaky Bucket shaper is needed.

3.4.1 Rate control

The Peak Cell Rate (PCR) is set with the **leak2** and **splash2** (of Leaky Bucket2) parameters according to:

$$\mathbf{PCR} = \frac{\mathbf{leak2}}{\mathbf{splash2}} \bullet \mathbf{systemrate}$$

The Sustainable Cell Rate (SCR) is set with the **leak1** and **splash1** (of Leaky Bucket1) parameters according to:

$$\mathbf{SCR} = \frac{\mathbf{leak1}}{\mathbf{splash1}} \bullet \mathbf{systemrate}$$

The splash is normally chosen in the range 128..255 because this gives the best accuracy. A rate change is then possible in steps of $\frac{1}{128}$ (accuracy 0.78%) to $\frac{1}{255}$ (accuracy 0.39%).

The minimum PCR and SCR can then be calculated from the minimum leak and maximum splash:

$$\frac{1}{128} \bullet \frac{1}{255} = \frac{1}{32640}.$$

With a linkrate of 155 Mbits/sec this gives a minimum rate of 4.7Kbits/sec.

The Maximum Burst Size (MBS) at PCR cells is determined by:

$$\mathbf{MBS} = \frac{\mathbf{limit1}}{\mathbf{splash1} - \frac{\mathbf{splash2}}{\mathbf{leak2}} \bullet \mathbf{leak1}} + \mathbf{1}$$

The CDV generated by the Leaky Bucket is determined by:

$$\mathbf{CDV} = \left\lfloor \frac{\mathbf{limit2}}{\mathbf{leak2}} \right\rfloor [celltime] \Rightarrow \mathbf{leak2} \bullet \mathbf{CDV} \le \mathbf{limit2} < (\mathbf{leak2} + \mathbf{1}) \bullet \mathbf{CDV}$$

The CDV is defined to be a positive integer in celltime units.

The minimum CDV is zero (limit = 0). The maximum CDV depends on the leak2 rate and the limit2. From the formula it can be seen that the largest CDV is reached with low leak2 rates or a high limit2.

If the leak rate and the CDV are given the necessary limit2 can be calculated by the formula:

$$\mathbf{limit2} = \lceil \mathbf{leak2} - \mathbf{1} \rceil + \lfloor \mathbf{leak2} \bullet \mathbf{CDV} \rfloor$$

Setting a CDV results in a maximum number of consecutive cells:

$$\mathbf{Maximum\ number\ of\ consecutive\ cells} = \frac{\mathbf{limit2}}{\mathbf{splash2} - \mathbf{leak2}} + \mathbf{1} \approx \frac{\mathbf{limit2}}{\mathbf{splash2}} + \mathbf{1} \quad (\mathbf{leak2} << \mathbf{splash2})$$

Note that CDV and the maximum number of consecutive can not be set independently.

The dual Leaky Bucket mode is specially used to control the MBS at PCR. As discussed above the burst from a single Leaky Bucket consists of consecutive cells. Thus the data is sent at the full link rate. In most cases this is not tolerable. To allow not only control of the PCR, but also of the MBS at PCR and the SCR, a second Leaky Bucket is used.

3.4.2 Examples

Assume Leaky bucket one has Splash1 = 10, Leak1 = 1, Limit1 = 9000. Thus it allows cells output at 1/10 of the link rate and outputs 1001 consecutive cells if possible. The output is shown in Figure 3-5.

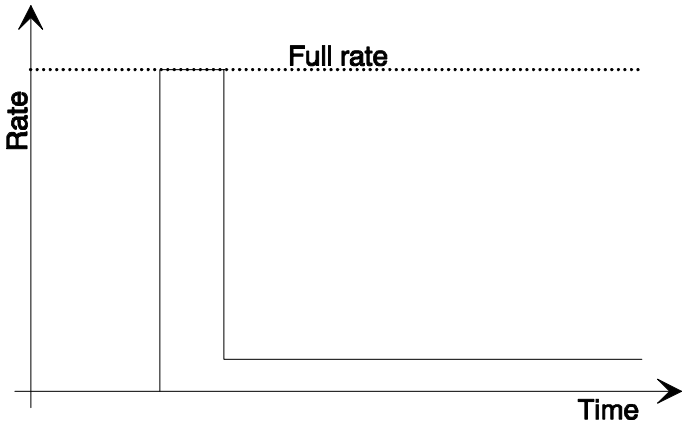


Figure 3-5: Single Leaky Bucket with burst

Assume Leaky bucket two has Splash2 = 5, leak2 = 1, Limit2 = 0. Thus it allows cells output at 1/5 of the link rate and has a CDV of zero. The dual Leaky Bucket function outputs a cell only if **both** Leaky Buckets say that it is OK. Under normal operating conditions bucket one will control the Sustainable Cell Rate (SCR) and Bucket two the Peak Cell Rate (PCR). However if the level of bucket one is zero it allows the output of 1001 consecutive cells. This is blocked by bucket two which allows a maximum rate of 1/5 of the link rate (PCR = 1/5). Thus instead of 1001 consecutive cells, the SHAP3 will output 1801 cells at a fifth of the link rate (MBS = 1801) and then output all other cells at a tenth of the link rate (SCR = 1/10). This is shown in Figure 3-6.

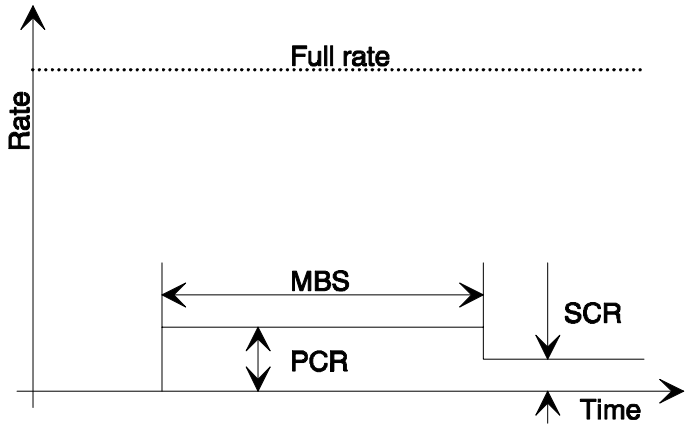


Figure 3-6: Dual Leaky Bucket without CDV

An even more complicated output function can be built by allowing a certain CDV. In such a case the SHAP3 will output number of consecutive cells at the full link rate, then go down for (MBS - number of consecutive cells) at a fifth of the link rate (PCR) and finally settle at a tenth of the link rate (SCR). This gives an output flow in three stages from which the height of stage two and three can be controlled. This is shown in the following figure:

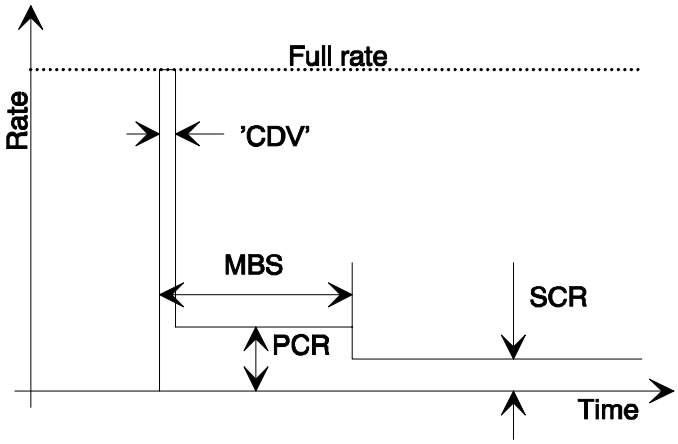


Figure 3-7: Dual Leaky Bucket with CDV

3.5 Cell Delay Variation

When setting the shaper parameters a user should keep in mind that the network can change the distance between cells. This is called the cell delay variation (CDV). Thus even if the SHAP3 outputs all the cells with equal distance, it is not guaranteed that they also keep this distance throughout the network. Especially if the cells arrive at the UPC in the network cells can be removed because the 'logic' between the SHAP3 and the UPC block in the network has changed the inter-cell distance.

4 Multiple classes

4.1 Priorities setting

It is possible that multiple classes perform a request to output a cell. As the SHAP3 has only a single output port a decision must be made about which class is allowed to sent a cell. The priority decoder is needed to perform such arbitration. The SHAP3 allows a mixture of two types of priorities: Static priority and Round Robin priority.

4.1.1 Static priority

Using Static priority gives a known behaviour of the SHAP3 towards cell-requests. The class with the lowest number has the highest priority. Thus class zero has priority over class one which has priority over class two etc. Normally at least two classes

are working in static priority because time-critical cells (For example from AAL1 type traffic) must be transported over the network with minimal delay.

4.1.2 Round Robin priority

In Round Robin priority the classes are polled one after the other if a cell-request is pending. If this is the case the selected class is allowed to send the cell. The next time the requests are started with the class following the one just served. After reaching class 31 class 0 is polled again. This mechanism allows multiple classes equal access to the output port.

Of course the polling of the 32 classes is all done in one cell time.

4.1.3 Mixed priorities

The following priority schemes are possible:

- All static
  Class 0 has highest priority. Class 31 has lowest priority.
- All round robin
  All classes have equal priority and are serviced on a round-robin basis.
- The classes 0..X have static priority, all other classes X..31 are round robin, but classes 0..X have priority over classes X..31.

Priority code	Static	Round-Robin
0	-	0..31
1	0..1	2..31
2	0..3	4..31
3	0..5	6..31
4	0..7	8..31
5	0..9	10..31
6	0..11	12..31
7	0..13	14..31
8	0..15	16..31
9	0..17	18..31
10	0..19	20..31
11	0..21	22..31
12	0..23	24..31
13	0..25	26..31
14	0..27	28..31
15	0..29	30..31
16	0..31	-

4.2 Queue output conflicts

The SHAP3 is normally used with more than one active class. As described above the classes will influence each other when cells should be output. If no measurements are taken some nasty side-effects can occur. Below we give an example of two classes where one is heavily distorted by the other.

We have one class sending cells at a ratio 1:4 (One cell in 4 time frames), CDV 0 which has a high priority and a second class sending cells at a ratio 1:3, CDV 0 with a lower priority. At some moment both want to output a cell at the same time however only one of them is allowed to do so. As the one with ratio 1:4 has the highest priority the other has to wait for one cell time. The result is shown in Figure 4-1.

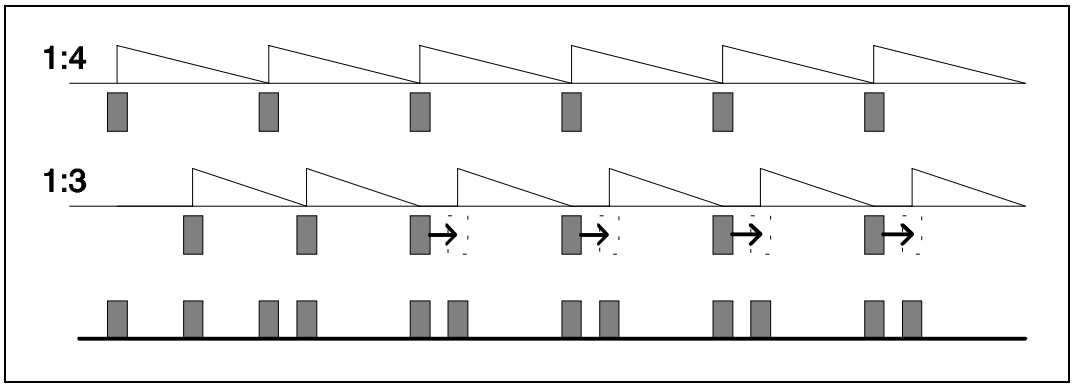


Figure 4-1: Cell output without CDV

Once a collision occurs the class with the lowest priority must wait before a cell can be output. In Figure 4-1 this is indicated with an arrow showing that the cell was shifted for one time slot. From this moment on every cell gives a collision as the ratio 1:3 with a time delay of one cell (due to the higher priority of the 1:4 class) also gives a ratio of 1:4. The net effect is that the lower priority class is sending at a ratio of 1:4 instead of 1:3. If the round-robin priority mechanism is used it gives the same result and does not help here.

A solution to the problem is to set a very small CDV of one for the low priority class. This is shown in Figure 4-2.

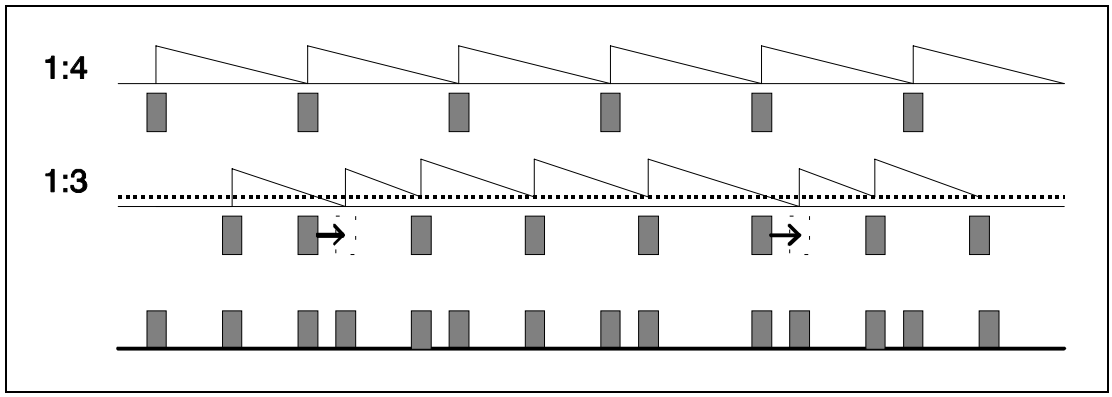


Figure 4-2: Cell output with CDV of one

Here the Leaky Bucket has a kind of 'memory' when a collision occurs and the cell had to be output with a small delay. Therefore the next cell is output a cellslot earlier (CDV of one).

5 Class data

Each class has a number of configuration and status values stored in three internal RAMs (Queue control RAM, Leaky Bucket parameter RAM and Leaky Bucket level RAM). The internal RAMs can not be accessed directly by the microprocessor. To write to these RAMs the microprocessor must first write the *all purpose data register* with the data and set the number of the class to write to. Then it must give a command to transfer this *all purpose data register* to the specified internal RAM. The opposite is valid for reading the class data. The microprocessor must first specify which class should be read and then give a command to transfer the RAM data to the *all purpose data register*.

All configuration data in all three RAMs should be initialised after reset!

5.1 Queue control RAM

5.1.1 Queue base address

The base address determines where the queue starts in the main memory. The start position is in **blocks** from the memory begin. One block holds 4096 words of 32 bits and can store upto 288 cells. The register is 13 bits wide. The first words in the main memory are used for the class look-up table. This table can be as small as 1 byte or as big as 64K bytes. Even if a single

byte is used this byte may not be overwritten by data. So at least one block must be reserved for the look-up table. The minimum queue start value is then 1. For a look-up table of 64K blocks 0..3 are used and the first free block is number 4.

Note:
A cell containing 53 or 56 bytes is stored in 14 words each 32 bits wide. One block of 4096 words can hold:

$$\frac{4096}{14} = 292 \text{ cells}$$

However a complete cell is stored and read-back in page mode[3]. As a consequence we can not utilise the memory for the full 100% and thus only 288 cells are stored in a space which **could** hold 292 cells. This gives an efficiency of almost 99%.

5.1.2 Queue size

The 21 bit wide queue size value determines the size of the queue in cells. This allows classes holding a maximum of 2M cells to be defined. Although the start addresses of the queues are 288 cells apart, the queue-size parameter allows smaller queues to be defined. As this is a loss of memory utilisation a minimum queue size of 288 cells is proposed.

The minimum queue size is 2 cells. Setting a smaller queue size will result in unpredictable behaviour of the SHAP3.

5.1.3 CLP-level

The value of the CLP level is compared with the queue level. If the queue level is above the CLP level incoming cells with the CLP bit set can be discarded if CLP discard is enabled.

To enable the CLP discard feature the CLP-cell discard enable bit in the *DRC register* must be set. Bit 7 of the *DRC register* controls the type of cells leaving the device:

- If bit 7 of the *DRC register* is cleared all cells are treated equal.
- If bit 7 of the *DRC register* is set cells with incoming CLP = 1 are discarded if queue level>CLP level.

If this bit is set the SHAP3 looks at the queue level for each incoming cell with the CLP bit set (CLP = 1). If the CLP bit is set AND the queue level is higher than the CLP level the cell will not be stored in the queue.

By setting the CLP-level equal to or larger than the queue size the CLP level becomes inactive. The CLP-level value is 21 bits wide.

The discarded cell counter can be programmed to count these discarded cells as lost or ignore these discarded cells. See also the next chapter.

5.1.4 Discarded cells counter

The discarded cell counter counts how many cells were discarded for a queue. If the maximum value has been reached it this value can be hold or it can set the interrupt bit and start counting from zero again.

- If bit 5 of the *all purpose data register* 1 is cleared when initialising the queue control RAM the counter counts to the maximum value and stays there.
- If bit 5 of the *all purpose data register* 1 is set when initialising the queue control RAM the counter rolls over when reaching the maximum value and sets a status bit.

The counter can also be programmed to count the discarded cells with CLP = 1.

- If bit 6 of the *all purpose data register* 1 is cleared it is also incremented when a cell arrives which has the CLP bit set and the queue level is above the CLP level.
- If bit 6 of the *all purpose data register* 1 is set when initialising the queue control RAM it is only incremented when the queue is full (queue overflow).

When cells are discarded due to an erroneous HEC or a FIFO full status, the counter is not incremented. The cell counter can not be written independent of the rest of the queue data. Thus it is not possible to reset the counters.

5.1.5 Queue level

[3]Page mode is a special means of operation of a dynamic RAM. It means a single RAS cycle followed by multiple CAS cycles. In page mode all CAS addresses must remain in a single RAS page. For more details see the data sheet of a dynamic RAM which can work in page mode.

This value can only be read. It shows how many cells are stored for that class. The queue level value is 21 bits wide and is automatically updated when a parameter set is read from the on chip queue control RAM.

5.1.6 Queue status bits

There are three status bits for each queue, the empty, full and CLP level status bit. These bits are automatically updated when a parameter is read from the on chip queue control RAM. The high-level status bit is set if the queue level is larger than the CLP-level.

5.1.7 Read/Write queue control RAM

The queue base address, queue size, CLP level, discarded cells counter, read pointer, write pointer and the three configuration bits (*all purpose data register* 1 bit 5 .. 7) are stored in the queue control RAM. It is not possible to access the queue control RAM directly from the microprocessor. Instead the *all purpose data register* is used for intermediate data storage. Under all conditions it is possible to address the queue control RAM.

5.1.7.1 Write queue control RAM

To write to the queue control RAM first the *all purpose data register* 0..16 must be written. It is important that the read pointer, write pointer and bit 7 of the *all purpose data register 1* used by the queue control algorithm are cleared when setting up a queue control entry. Then the *class select register* must be set with the number of the class to write to. Finally the command *transfer to queue control RAM* (code 0x03) must be written to the *transfer control register*. The SHAP3 will then update the queue control RAM in such a way that it does not interfere with the normal operations. Bit 7 of the *transfer control register* shows the status of the transfer:

- If bit 7 of the *transfer control register* is cleared the transfer is still pending. No new data may be written to the *all purpose data register* while a transfer is pending.
- If bit 7 of the *transfer control register* is set the transfer is ready.

5.1.7.2 Read queue control RAM

To read the data from the queue control RAM the first task of the microprocessor is to set the *class select register* with the number of the class to read. The command *transfer from queue control RAM* (code 0x02) must be written to the *transfer control register*. The SHAP3 will then read the queue control RAM data and transfer it to the *all purpose data register*. Bit 7 of the *transfer control register* shows the status of the transfer:

- If bit 7 of the *transfer control register* is cleared the transfer is still pending. The data in the *all purpose data register* is not yet valid
- If bit 7 of the *transfer control register* is set the transfer is ready. The data in the *all purpose data register* is valid

Only after the transfer is completed the microprocessor reads the values from the *all purpose data register*.

5.2 Read/Write Shaping settings

The shaping settings are stored in two RAMs:

- The Leaky Bucket parameter RAM
- The Leaky Bucket level RAM

Both RAMs can be set independently. It is allowed to change the Leaky Bucket settings while the SHAP3 is active.

All shaping parameters for all classes whether they are used or not should be initialised after reset! If a class is not used the related shaping parameters should be set to not OK (see chapter3.2.2 Stop data stream)

5.2.1 Write parameter or level RAM

To write to the RAM the *all purpose data register* must be written first. For the parameters it is the register 0-9, for the level it is the register 0-7. Then the *class select register* must be set with the number of the class to write to. Finally the command *transfer to parameter RAM* (code 0x05) or *transfer to level RAM* (code 0x15) must be written to the *transfer control register*. The SHAP3 will then update the corresponding RAM in such a way that it does not interfere with the normal operations. Bit 7 of the *transfer control register* shows the status of the transfer:

- If bit 7 of the *transfer control register* is cleared the transfer is still pending. No new data may be written to the *all purpose data register* while a transfer is pending.
- If bit 7 of the *transfer control register* is set the transfer is ready.

5.2.2 Read parameter or level RAM

To read the data from the Leaky Bucket parameter or level RAM the first task of the microprocessor is to set the *class select register* with the number of the class to read. The command *transfer from parameter RAM* (code 0x04) or the command *transfer from level RAM* (code 0x14) must be written to the *transfer control register*. The SHAP3 will then read the corresponding RAM data and transfer it to the *all purpose data register*. Bit 7 of the *transfer control register* shows the status of the transfer:

- If bit 7 of the *transfer control register* is cleared the transfer is still pending. The data in the *all purpose data register* is not yet valid.
- If bit 7 of the *transfer control register* is set the transfer is ready. The data in the *all purpose data register* is valid.

Only after the transfer is complete the microprocessor can read the values from the *all purpose data register*.

5.2.3 Read/write static RAM

5.2.3.1 Read/write parameter set

When reading parameter sets from or writing to the external static RAM the procedure is almost the same as described above.

Two differences are to be observed:

1. Besides the *class select register* also the *parameter set control register* must be set with the parameter set number.
2. The transfer codes (was 0x05 & 0x04) should be 0x07 and 0x06.

Bit 7 of the *parameter set control register* must be set. Otherwise not the parameter sets but the pointer to the parameter set is accessed.

5.2.3.2 Read/write parameter set pointer

When reading the parameter set pointer from or writing to the external static RAM the procedure is almost the same as described in chapter 5.2.3.1 Read/write parameter set.

1. Select a class using the *class select register.*
2. Write the parameter set pointer value to the *all purpose data register* 9 (only necessary when writing).
3. Clear bit 7 of the *parameter set control register.*
4. The transfer codes should be 0x07 (for writing) and 0x06 (for reading).

After finishing the read transfer the *all purpose data register 9* holds the parameter set pointer value.

6 CLP bit handling

The device can handle cells with and without CLP bit set differently. This is done by setting an CLP-level for a class and setting the CLP-discard bit in the *DRC register*. As soon as the queue level is beyond the CLP-level all arriving cells which have the CLP bit set are removed. The counter mode determines if these cells are also counted as discarded cells. The SHAP3 is not capable to remove cells with the CLP bit set that are already stored in the queue.

The CLP bit handling can be disabled for a certain class by filling the CLP-level with a value equal to or larger than the queue size. A general CLP enable bit must be set in order to handle cells with incoming CLP-bit set specially.

Bit 7 of the *DRC register* is a global CLP-cell discard enable/disable bit.

- If bit 7 of the *DRC register* is set the CLP level compare for all classes is enabled
- If bit 7 of the *DRC register* is cleared the CLP level compare for all classes is disabled

7 Class assignment

Arriving cells are processed on basis of the cell header to assign them to a class. To do this the SHAP3 uses a class lookup table with maximum 64K entries. From each arriving cell up to 16 bits are taken from the header (See *header mask register*). The value of these bits is taken as address in the lookup table. From the lookup table a byte is read. The LS 5 bits of this byte specify the class number where the cell should be written to.

The class lookup table is stored in the external dynamic memory. The size of the class look-up table depends on the number of bits set in the *header mask register*. The maximum number of bits set is 16. For 16 bits the SHAP3 can address 2^{16}=64K entries. Thus the look-up table must be 64K bytes large (4 blocks each 16K bytes). If less then 16 bits are set the logic will force all non-used bits to zero. As a result the MS address bits will always be zero and thus the lookup table can be smaller. The table below gives the size of the look-up table in bytes and SHAP3 memory blocks against the number of header extraction bits.

Header extr. bits	Table size	Blocks	First free Block
16	64K	4	4
15	32K	2	2
14	16K	1	1
13	8K	1	1
12-0	4K	1	1

Even for the smallest header extraction value (0 bits) you need at least 1 block. Thus below 14 header extraction bits there is no more gain for memory.

Note that if you set the header extaction mask to all zero's the result will always be zero. In that case only a single byte in the memory (address 0) will be used. Thus all cells go into the queue specified at address 0 of the lookup table.

7.1 Header mask

The header mask is used to extract up to 16 bits of the first four octets of a cell. The extracted bits are presented to the class lookup table. The header bits are selected using the four byte *header mask register*. Each bit set in a register will cause the corresponding header bit to be used for selecting a class. If less than 16 bits are set in the mask, the remaining extracted bits are set to zero. If more than 16 bits are set only the last 16 bits are used. *Header mask register* 0 selects bits from octet 1, register 1 from octet 2, etc. The LS bit of a register (bit 0) selects data arriving at input RxData(0).

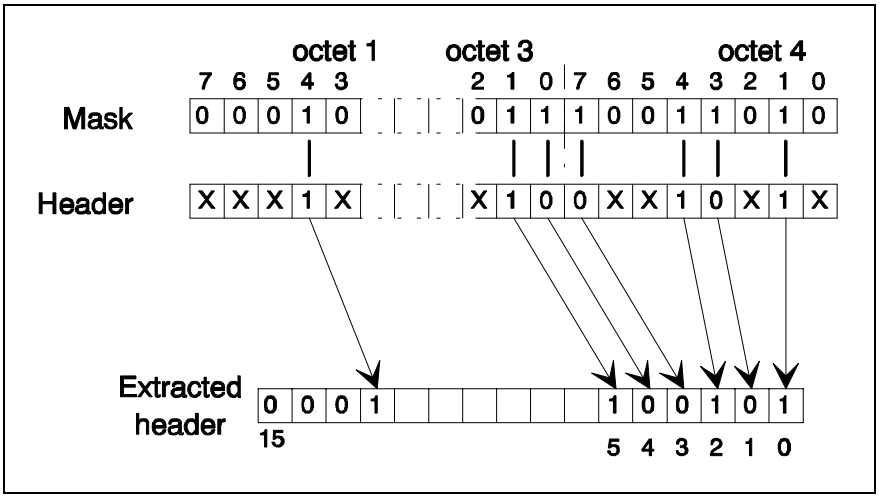


Figure 7-1: Header extraction mask

7.2 Lookup table

The size of the lookup table can vary between 1 byte and 64Kbytes. As the memory of the SHAP3 is allocated in blocks of 4096 words, the smallest lookup table requires 1 block of 4K words the largest requires 4 blocks of 4K words. The memory connected to the SHAP3 should be large enough to hold the lookup table and still have enough storage left to store the cells from the classes.

The maximum extracted header value of 16 bits is only present in excessive circumstances. In most cases the extracted header value will be smaller and thus less memory is needed for the lookup table. However if the complete 16 bits are essential the SHAP3 only requires enough DRAM to fulfil the demands.

If less than 15 bits are used only a single block is required to hold the complete lookup table. Note that if no header extraction bits are set the extraction logic always produces the value 0. Thus the first byte in the memory is used to determine to which class the cell belongs. However as the memory is allocated in blocks, 1 block (4K words) is the minimum lookup table size which is always required.

The SHAP3 can transfer data to and from the class lookup table. The microprocessor has no direct access towards the table. Even if a link is fully loaded (no gaps or idle cells) it is possible to access the lookup table. The lookup table is in the dynamic RAM and each word is 32 bits wide. As for a class we need only 5 bits (as we have only 32 classes) one word holds 4 entries: the LS 5 bits of each byte. Therefore a read or write access will always read/write 4 consecutive table entries.

Unused classes should not have an entry in the lookup table.

As the size of the lookup table is variable the number of address bits used to access the lookup table is also variable. The number of address bits equals the number of bit set in the *header mask register* minus 2. Setting a wrong address during a read or write access will cause the SHAP3 to read or write at the wrong place in the memory. When writing data this can cause the SHAP3 to output damaged cells.

To update the lookup table the device has two registers:

Class lookup table address

This value is the address to access. It is 32 bits wide (all purpose data register 4 .. 7) where only the least X significant bits should be used. X equals the number of bit set in the header mask register minus 2. The most significant bits should be cleared for lookup table access.

Class lookup table data

This value is the data to be read/written (all purpose data register 0 .. 3). From each byte the LS 5 bits are used to indicate one of the 32 queues. Byte 0 of word 0 corresponds with an extracted header value of 0x0000. The second byte of the first word with extracted header value of 0x0001, etc.

To write towards the lookup table, the four class entries (*all purpose data register* 0-3) and the destination address (*all purpose data register* 4-7) are written into the SHAP3. A write transfer will start after writing 0x01 to the *transfer control register*. To read from the lookup table, the source address is written into the SHAP3 (*all purpose data register* 4-7). A read transfer will start after writing 0x00 to the *transfer control register*. The SHAP3 will load the data from the table into it's registers. In both cases the transfer ready bit must be observed to prevent errors.

With a 19.44MHz clock a transfer will take maximum 5.4 µsec (53 byte cell format).

7.3 General memory access

In the previous chapter we concentrated on reading from or writing to the lookup table. Therefore we restricted the number of address bits which should be set.

However more than the LS 14 address bits of the class *lookup table address register* can be used. This will only cause the SHAP3 to access a different memory location.

The same holds for the data registers. For the lookup table all 32 bits can be set but only the LS 5 bits are used. For a general memory access all the 32 bits are valid.

The commands to read or write the memory are the same. General memory access can be used to perform a memory check on the DRAM or to access the DRAM in a random way. It is only dangerous to do so when the SHAP3 is operating as it may inadvertently destroy data stored in the memory.

8 Class Rate Adaptation

Often the rate at which data is sent must be changed. This is what we call rate adaptation, which means that the output rate parameters for the Leaky Bucket function can be changed during runtime either under software or under hardware control. This function is available in two forms:

- With external static RAM 32 rates are available per class per Leaky Bucket.
- Without external static RAM 2 rates are available per class if working in single Leaky Bucket mode.

8.1 Leaky Bucket select register

When working in single Leaky Bucket mode (controlled by bit 7 of the *Leaky Bucket control register*) the SHAP3 still has two single Leaky Bucket parameters per class available. This register determines for each class whether single Leaky Bucket one or two is used.

8.2 Parameter set select

One parameter set consists of either one dual Leaky Bucket parameter in dual Leaky Bucket mode or two single Leaky Bucket parameters in single Leaky Bucket mode. As mentioned rate adaptation allows a user to choose one of 32 available output rates per class. The output rate is stored in an external static RAM as 32 Leaky Bucket parameter sets for each of the 32 classes. Therefore total 1024 parameter sets are stored in the external static RAM. Only one parameter set can be active for a class. Selecting a parameter set can be done under software or hardware control. For each class a 5 bit parameter set pointer is also stored in the external static RAM which points to the active parameter set. Hence 32 pointers are stored. This pointer indicates the number of the active parameter set.

For example:

To select parameter set 5 the value of the pointer should be set to 0x5.

All parameter sets and all pointers in the external static RAM can be changed under CPU control (see chapter 5.2 Read/Write Shaping settings). For each class the actual parameters are stored in the parameter RAM of the SHAP3. The transfer of parameters from the external RAM to the internal RAM can be performed in two ways:

- If bit 4 of the *cell out control register* is cleared the new parameter set can be loaded under hardware control.
- If bit 4 of the *cell out control register* is set the new parameter set can be loaded software controlled.

Two operations are possible to change the active parameter set: **increment** and **decrement** commands can be used to change the pointer value.

The increment always chooses the next higher rate. For the decrement four different decrement modes are available:

- Decrement by 1
- Decrement by 25%
- Decrement by 50%
- Decrement by 75%

The decrement rate is programmed with bits 5 and 6 of the *parameter set control register*:

Bits [6:5]	Decrement by
00	1
01	25%
10	50%
11	75%

If the lowest/highest parameter set is selected the decrement/increment command has no effect. The conversion from the various decrement modes are shown in the conversion table below.

The conversion of the **old** parameter set pointer to the **new** parameter set pointer depends on the decrement **mode bits [6:5]** in the *parameter set control register*. The mode bits 6 & 5 are shown below the "New" text:

Old	New 00	New 01	New 10	New 11	Old	New 00	New 01	New 10	New 11
0	0	0	0	0	16	15	12	8	4
1	0	0	0	0	17	16	13	8	4
2	1	1	1	0	18	17	14	9	4
3	2	2	1	0	19	18	15	9	4
4	3	3	2	1	20	19	15	10	5
5	4	4	2	1	21	20	16	10	5
6	5	5	3	1	22	21	17	11	5
7	6	6	3	1	23	22	18	11	5
8	7	6	4	2	24	23	18	12	6
9	8	7	4	2	25	24	19	12	6
10	9	8	5	2	26	25	20	13	6
11	10	9	5	2	27	26	21	13	6
12	11	9	6	3	28	27	21	14	7
13	12	10	6	3	29	28	23	14	7
14	13	11	7	3	30	29	23	15	7
15	14	12	7	3	31	30	24	15	7

The new value of the parameter set pointer is used to select the new parameter set in the external static RAM. The SHAP3 transfers this new parameter set from the external to the internal RAM. If the highest/lowest class is active further increment/decrement requests are ignored.

8.3 Software rate control

To control the output rate under software there are two possibilities:

- Write new parameters to the internal RAM.
- Transfer parameters from the external static ram to the internal parameter RAM using the increment or decrement command.

The first method is described above and although it allows an arbitrary rate to be set it also requires that the microprocessor sets all the registers and transfers the registers to the parameter RAM.

To transfer parameters from the external static RAM to the internal parameter RAM the increment and decrement commands can be performed under software control. Bit 4 of the *cell out control register* must be set for software control. To increment/decrement the pointer three microprocessor write actions must be performed:

1. Write the class number in the *class select register*, the inc/dec bits must be zero.
2. Write the class number plus the increment/decrement bit in the *class select register*. The parameter set pointer of the class is incremented/decremented. The new value of the parameter set pointer is used to select the new parameter set in the external static RAM. The SHAP3 transfers this parameter set from external to the on chip parameter RAM.
3. Write the class number in the *class select register* and set the inc/dec bits zero again.

It is also possible to set a new pointer value for the parameters under software control. However this command sets only the pointer but does **not** load a new parameter set. An increment or decrement request is always handled in one cell time (53 or 56 cycles). The next increment or decrement command must be at least 53 (or 56) clock cycles after the first pulse. If the next command is less than 53 (or 56) clock cycles after the first one, this additional command may be accepted or not, depending on the internal state of the state machine.

8.4 Hardware rate control

To control the output rate using external hardware control the SHAP3 has seven input signals:

class [4:0]:

5 bits wide to select one of the 32 classes.

Par_inc:

The rising edge of this signal stores the selected class and increments the parameter set pointer of this class. The new value of the parameter set pointer is used to select the new parameter set in the external static RAM. The SHAP3 transfers this parameter set from external to the on chip parameter RAM.

Par_dec:

The rising edge of this signal stores the selected class and decrements the parameter set pointer of this class, depending on one of the above decrement formulas. The new value of the parameter set pointer is used to select the new parameter set in the external static RAM. The SHAP3 transfers this parameter set from external to the on chip parameter RAM.

With this method it is not possible to directly select one of the 32 available rates. Instead of this the next higher or a lower rate (depending on the selected decrement formula!) is chosen[4] Signal timing:

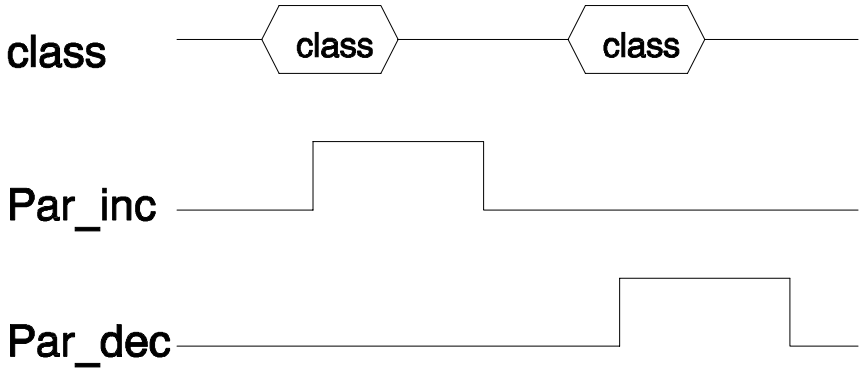
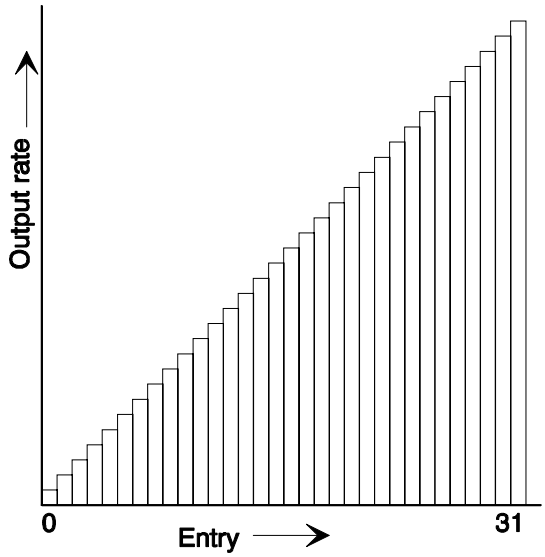


Figure 8-1: Par_inc, Par_dec signal timing

When an PAR_inc or PAR_dec pulse arrives this event is stored together with the class number until the chip can execute this command. An increment or decrement request is always handled in one cell time (53 or 56 cycles). The next PAR_inc or PAR_dec pulse must be at least 53 (or 56) clock cycles after the first pulse. If the next pulse is less than 53 (or 56) clock cycles after the first one this additional pulse may be accepted or not depending on the internal state of the state machine.

8.5 Parameter curves

Because of the increment-decrement signals the parameter sets in the table should be ordered in rising or falling output rate. However how fast rising or falling is up to the user. Some possible configurations are:



[4]Assuming the rates are stored from low rate to high rate.

Figure 8-2: Linear rate change

The linear rate change is the most likely to be used. It will cause the SHAP3 to go up or down in a regular way.

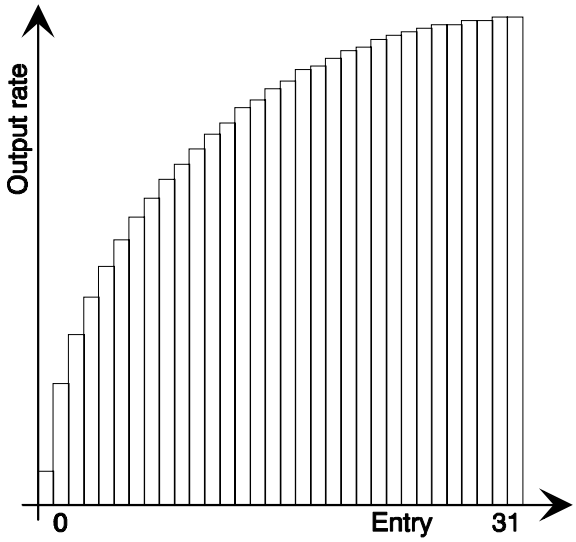


Figure 8-3: Logarithmic rate change

The logarithmic rising range allows a rate increment and decrement to have small change at higher rates and a big change at low rates.

The complement from this function: *logarithmic falling* is not shown here but the reader can easily imagine what it looks like. It allows a rate increment and decrement to have large changes at higher rates and a small change at low rates.

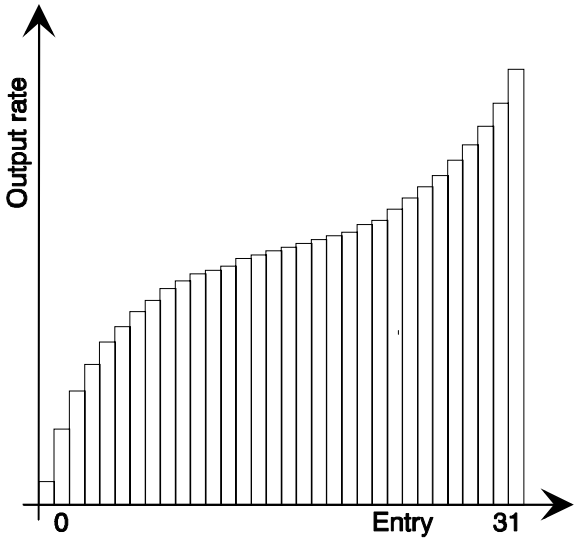


Figure 8-4: Pi-Curve rate change

The PI-curve allows changes in the middle to have lilt effect and at the low and high ends to have more impact.

An alternative to this is the S-curve which is not shown here. It causes changes in the middle of the curve to have a large impact and increments/decrements at the extreme ends to give only little changes.

It is also possible to fill all entries with the same data effectively rendering the parameter change inoperable.

8.5.1 Asymmetric change (ABR)

Thus far we handled the increment and decrement equal. As mentioned above the SHAP3 has four modes in which a decrement can occur:

- Decrement by one.
- Decrement by 25%
- Decrement by 50%
- Decrement by 75%

This option allows the SHAP3 to be used in systems which require ABR or any other requirement of fast rate reduction. An output reduction of 50% means the SHAP3 will select the parameter set which is 50% below the current one. This does not mean a reduction of 50% of the output rate as the output rate depends on the filling of the parameter table. Only if the parameter table is set up linearly will a decrement of 50% also give an output rate reduction of 50%. Further more the reduction can slightly differ as only 32 parameter sets are available. E.g. if parameter set 3 is active a reduction of 50% can not be performed (there is no parameter set 1.5).

This option can be combined with the parameter curves described above allowing a large and flexible range of output rates.

The increment parameter always works with an increment of one.

9 Local cells transfer

Local cell transfer is a way to access cells in the SHAP3 memory. For a normal passing cell stream the SHAP3 does not allow cells to be copied from or to the memory. This is due to the fact that the memory pointers can change between microprocessor reads or writes. Hence it is only possible to do this for classes that do not output cells (process incoming cells only) or which do not input cells (process outgoing cells only).

Reading or writing of cell data can be done under CPU control or under DMA control.

As the SHAP3 is not primarily intended to be used as starting point or ending point for data, the transfer rate of cells from or to the memory is low due to several factors:

- The data port is only 8 bits wide.
- Data transfer to or from the memory can happen worst case only once every 6 µsec.
- No built-in DMA controller.

However these options allow a user to inspect data in the memory, debug a stream or use a SHAP3 class as AAL device for low speed data. If really high speed data transfer to or from a microprocessor is needed the user can add a logic to the input port or to the output port of the device:

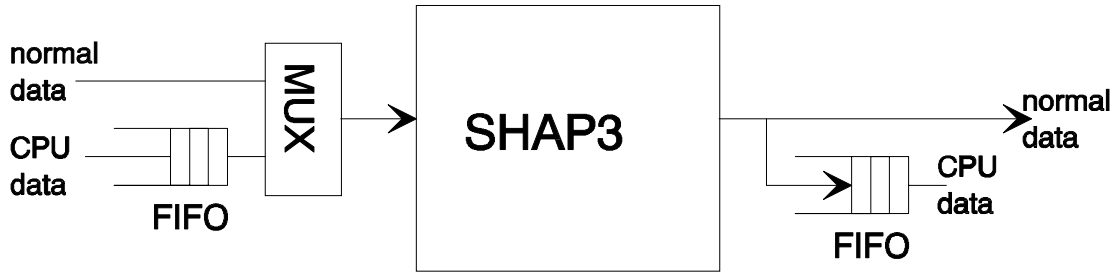


Figure 9-1: Faster access to SHAP3 data

This logic allows the microprocessor to write towards the SHAP3 memory using an external FIFO at the input port. Data speeds up to 200 Mbit/sec can then be processed. The same holds for reading data from the SHAP3 memory. A FIFO at the output port connected to the CPU data bus allows a user to read data up to 200 Mbit/sec from the SHAP3 memory. If the data is written in 53 byte format and read back in 56 byte format the data coming from the output port will contain the 24 bit time stamp added to the cell. Note that the rate at which cells are forwarded to the CPU can be controlled by selecting a certain output rate. This allows a user to store high speed arriving data at the input port and simultaneous start performing analysis on the data coming from the output port but at a lower rate. So the maximum number of analysed cells can be increased.

9.1 CPU cell handling

When transferring cells from or to the SHAP3 memory under CPU control it can be done on polling basis or under interrupt control. At all times a complete cell must be read or written before a class switch is made. Thus once the transfer for a cell has started it must be finished for that class. In the mean time another class may not be selected by the microprocessor.

The SHAP3 has status bits which signals the status of the transfer and the cell interface unit.

9.2 DMA cell handling

When transferring cells from or to the SHAP3 memory under DMA control it can be done for one class only at the time. Thus once the transfer for a cell has started it must be finished before another class is serviced by the DMA controller. The SHAP3 has a DMAREQ* signal that is directly connected to bit 7 of the *transfer control register*. Therefore all transfers can be done by polling bit 7 of the transfer control register or under DMA control using the DMAREQ* signal.

Note that the SHAP3 has no build-in DMA controller. To perform access with a DMA controller extra logic is needed to put the chip in read mode for the three general purpose data registers during a DMA access.

9.3 Read cell from data stream

Cells can be read from the class queue only if no cells are output. The *Leaky Bucket* parameter must be initialised with values that always will result in a '**not OK**' decision (see 0 So the Level is always lower than or equal to the Limit. In dual Leaky Bucket mode both parameters should be programmed this way.

Stop data stream). The following actions must be performed to read a cell from a class queue.

First the class must be set in the *class select register*.

A cell read transfer is started by writing 0x08 to the *transfer control register*. The cell interface unit then calculates and loads the read address of the queue of the selected class. Using this address the first four bytes of the cell are transferred to the *all purpose data register* 0..3. To see if the transfer has finished the microprocessor must check the status bit. When status bit 7 of the *transfer control register* is set the transfer is finished.

The first four bytes can now be read from the *all purpose data register* 0..3. The transfer of the next four bytes is started when reading the *all purpose data register* 3. The microprocessor must again wait until bit 7 of the *transfer control register* is set. Then the next four bytes are available in the *all purpose data register* 0..3.

This cycle is done fourteen times, until the complete cell (53 or 56 bytes) is read. Bit 6 of the *transfer control register* shows the status of the cell interface unit. When all bytes have been read bit 6 of the *transfer control register* is set.

9.4 Insert cell in data stream

Cells can be written to the class queue only if no cells arrive at the input for the queue. To accomplish this the class index may not be present in any entry of the class lookup table. The cell is output according to the Shaping function (unless the microprocessor writes cells in slower as the shaper function outputs). The following actions must be performed to write a cell to a class queue.

First the class must be set in the class select register.

The cell transfer is started by writing 0x09 to the *transfer control register*. The cell interface unit then calculates and loads the address of the queue of the selected class. To see if the calculation is finished the microprocessor must check the status bit. When status bit 7 of the *transfer control register* is set the calculation is finished.

The first four bytes of the cell can now be written to the *all purpose data registers* 0..3. The transfer of these bytes is started when writing the *all purpose data register* 3. The microprocessor must again wait until bit 7 of the *transfer control register* is set. Then the next four bytes can be written to the *all purpose data register* 0..3.

This cycle is done fourteen times, until the complete cell (53 or 56 bytes) is written. If the 53 byte cell format is used a dummy write to *all purpose data register* 3 is needed to start the fourteenth transfer. Bit 6 of the *transfer control register* shows the status of the cell interface unit. When all bytes have been written bit 6 of the *transfer control register* is set.

9.5 Cell interface transfer format

How the information is stored in the memory depends on the cell format programmed in the *input control register*. As the input controller supports three types of cells, three different memory formats are possible: 53 bytes cells, 53 bytes cells with 3 bytes tag where the cell header starts with the fourth byte, and 56 bytes cells. The input can also perform conversion from 56 byte cell format to 53 byte cell format. In this mode the first three bytes of a cell are removed.

When operating in 53 byte format or in conversion mode 56 to 53 byte format the cell data is temporarily stored in the *all purpose data register* according:

all purpose data register

reg 0	reg 1	reg 2	reg 3	
header 0	header 1	header 2	header 3	0
HEC	payload 0	payload 1	payload 2	1
payload 3	payload 4	payload 5	payload 6	2
				3
...				
payload 43	payload 44	payload 45	payload 46	12
payload 47	LS byte **24 bit time stamp**		MS byte	13

Figure 9-2: All purpose data register format 53 bytes

If storing 56 byte cells the information is stored as for 53 byte cells but the last 3 bytes contain cell data.When operating in 53 byte cell format plus 3 bytes tag the cell data format is:

all purpose data register

reg 0	reg 1	reg 2	reg 3	
tag 0	tag 1	tag 2	header 0	0
header 1	header 2	header 3	HEC	1
payload 0	payload 1	payload 2	payload 3	2
				3
...				
payload 40	payload 41	payload 42	payload 43	12
payload 44	payload 45	payload 46	payload 47	13

Figure 9-3: All purpose data register format 53 bytes + routing tag

10 Traffic receiver & generator

The SHAP3 is capable of sending or receiving a large number of cells in real time. Therefore it can be used as traffic analyser and as traffic generator. To support these functions some additional features have been built in the SHAP3.

10.1 Traffic receiver

The SHAP3 itself does not perform cell analysis. However it can store a large number of cells in real time. After storing the cells a microprocessor can read them from the memory and perform analysis of the received data. If using the maximum amount of memory and running at maximum speed (every arriving cell is a data cell) the SHAP3 can store data for 5.71 seconds. As extra feature the input stream can be split and assigned to various classes thus providing some means of filtering of the received cells. It is possible to assign all cells which do not need to be analysed to a small queue. This queue will overflow very fast, but as the information is not needed this is no problem.

During analysis it is often needed to know the arrival time of a cell. From the arrival time delay time, interarrival time and other useful information can be derived. The ATM cell is stored in 14 words of 32 bits in the dynamic memory. With a cell of 53 bytes, 3 bytes are unused. These bytes are used when the input controller is running in 56 bytes cell mode (cell type *input control register* bit [6:5] = "11" or "10"). In all other modes (cell type = "00" or "01") the 3 unused bytes are used for a 3 byte time stamp. The cell input controller of the SHAP3 has a 24 bit time stamp counter. This counter is incremented every RxClk cycle or every 53 (or 56) RxClk cycles. This counter value is written at the end of the cell when the cell type is "00" or "01". This feature can be used to measure the arrival time of the cells of a class. The Leaky Bucket parameters for that class can be programmed that no cell will leave the SHAP3 (always not OK, see chapter 3.2.2 Stop data stream). So all arriving cells of this class are stored in the queue. These cells can be read including the time stamp (see chapter 9.3 Read cell from data stream). The counter overflows after 2^{24} cycles and thus a minimum cell distance is required.

The time stamp increment every RxClk gives a very good resolution and can be used for discontinuous cell stream to measure the distance between two cells in RxClk. The disadvantage is that the minimum cell distance is a factor 53 (or 56) smaller.

The time stamp increment every 53 (or 56) RxClk cycles gives a low resolution and can be used for a continuous cell stream to measure the distance between two cells in 'cell times'. In this mode at least one cell every 45 seconds is required.

Note that the counter is running on the RxClk and for reliable measurements a continuos running RxClk is necessary.

10.2 Traffic generator

Of course the SHAP3 can function as traffic generator. A simple method of traffic generation is to write a large number of cells in a class and then start the shaping function. The duration of the data output depends on the output rate. Even with a limited amount of dynamic memory a user can produce traffic for a long period but only at a low rate. If using the maximum amount of memory and running at maximum speed the SHAP3 can produce 155.52 Mbit/s for 5.7 seconds.

A special option is built into the SHAP3 which is called "*circular output mode*". In this mode the data from a class is output continuously from the first to the last cell. To use this function the user must perform the following steps:

1. Disable the output of the class (see chapter 3.2.2 Stop data stream)
2. Set up the queue of the class for the number of cells to be sent.
3. Fill the **complete** queue with the pattern of cells which should be output.
4. Set the class in *circular output mode* (using bits 5 and 6 of the *cell out control register*)
5. Start the cell output by filling the Shaping parameters.

The SHAP3 will now output the data stream with the specified rate. Every cell in the queue is output and when the last cell is output the SHAP3 starts with the first cell of the queue again. This mode of traffic generation can be used for testing purposes but can also function very well as background traffic generator.

Using the method described first it is also possible to use all classes in non-circular mode to produce series of cells which are automatically output by the SHAP3 as soon as class zero does not produce data.

An even more complex traffic generator can be build using the static RAM to store different traffic rates and then switching between different rates for the various classes.

Bits 5 and 6 of the *COLB register* control the circular modes:

COLB [6:5]	Operation mode
00	Normal operation, All queues in normal mode.
01	Queue 0 in circular mode, Queue 1..31 in normal mode.
10	Queue 8..23 in circular mode, Queue 0..7 and 24..31 in normal mode.
11	All queues in circular mode

Those classes which operate in circular mode can only output traffic. As the queue is full they can not receive any incoming traffic. Those classes which do not operate in circular mode can perform the normal traffic shaping operations.

11 External memory

11.1 Dynamic memory

The SHAP3 can work with various types of dynamic memory. Not all of them are currently available but with the rate dynamic RAMs develop they will be in the future. The SHAP3 can work with one or two memory banks. How many banks should be used depends on the storage capacity a user wants to assign to the SHAP3.

11.1.1 Memory type

The SHAP3 can handle four different types of dynamic memory chips:

- 256K
- 1M
- 4M
- 16M

The number of the row and column address bits should be equal (symmetric DRAMs). It is also possible to use hyper page mode (EDO) DRAMs.

dram type	access time	system clk frequency
any	70 ns	up to 19.44 MHz
any	60 ns	up to 25 MHz

For optimal performance Bit 5 of the *dynamic ram control register* should be set (no additional CAS delay*)*.

The data bus width of the dynamic RAM is 32 bits. Thus dependant on the chip type you need several memory chips.

For example:

If using 256Kx8 bit chips, 4 chips are needed, if you using 1Mx1 bit chips 32 chips are needed.

The RAM modules as shown in the pictures below do not reflect all these possibilities but are drawn 'generic', i.e. without details about the internal configuration. The DRAM chip type must be programmed in the *dynamic ram control register* of the SHAP3.

The SHAP3 can use one or two memory banks with each of the above mentioned chips. This gives eight possible memory configurations:

1. 1 bank of 256Kx32 bits (1M bytes, 17K cells)
2. 2 banks of 256Kx32 bits (2M bytes, 37K cells)
3. 1 bank of 1Mx32 bits (4M bytes, 72K cells)
4. 2 banks of 1Mx32 bits (8M bytes, 148K cells)
5. 1 bank of 4Mx32 bits (16M bytes, 293K cells)
6. 2 banks of 4Mx32 bits (32M bytes, 593K cells)
7. 1 bank of 16Mx32 bits (64M bytes, 1.1M cells)
8. 2 banks of 16Mx32 bits (128M bytes, 2.3M cells)

The pin-layout of the SHAP3 is such that it can be connected to standard SIMM[5] modules without crossing wires.

It is up to the user to program the queues such that the queue is inside the available memory. The SHAP3 does not perform any checks to see if the queue is set up wrongly.

[5]That is, to **most** SIMM modules as there are so many different ones.

11.1.2 Refresh rate

Besides the type of RAM also the refresh rate can be programmed. The SHAP3 has one time slot per cell to perform either a refresh or a transfer to or from the memory. If possible the refresh rate should be programmed as low as possible as this increases the number of transfers that can be made. Note that if a refresh once per cell time is programmed the SHAP3 has no determined means to access the memory. Instead it must wait for a time slot where no cell arrives or leaves the memory to perform a transfer. The SHAP3 has a built in refresh counter and uses the RAS only refresh mode that is supported by all dynamic RAMs. Both banks are refreshed simultaneously The refresh time of the dynamic RAM can be set in the *dynamic ram control register*:

The refresh cycle time is given by the following formula:

$$\text{refresh cycle time} = \frac{\text{refresh\_num} \cdot \text{bytes\_per\_cell}}{\text{clk\_freq}}$$

clk_freq : system clock frequency

bytes_per_cell : 53/56 depending on bit1 of the *cell out control register*

refresh_num : refresh number selected by the *dynamic ram control register*

For example:

clk_freq = 20 MHz
bytes_per_cell = 53
refresh_num = 6

refresh cycle time = 1/20 MHz • 6 •53 = 15.9 E-06 sec
with a dynamic ram of 1Mx32, 1024 rows:
1024 • 15.9 E-06 s = 16.28 ms

which means 1024 refresh cycles every 16.28 ms

11.1.3 Configurations

The value of x in Figure 11-1 and Figure 11-2 depends on the memory type:

dram type	x
256Kx32	8
1Mx32	9
4Mx32	10
16Mx32	11

If the DRAM has an OE* pin it should be connected to GND.

11.1.3.1 Configuration with one memory bank

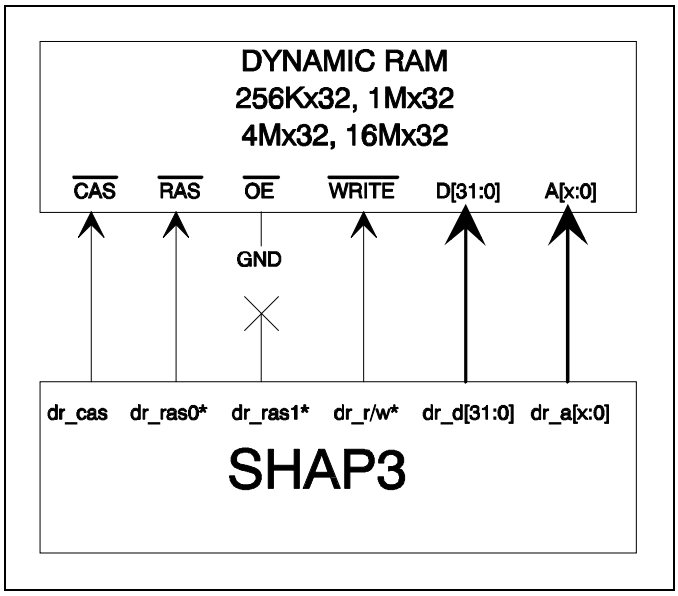


Figure 11-1: Memory configuration with one bank

11.1.3.2 Configuration with two memory banks

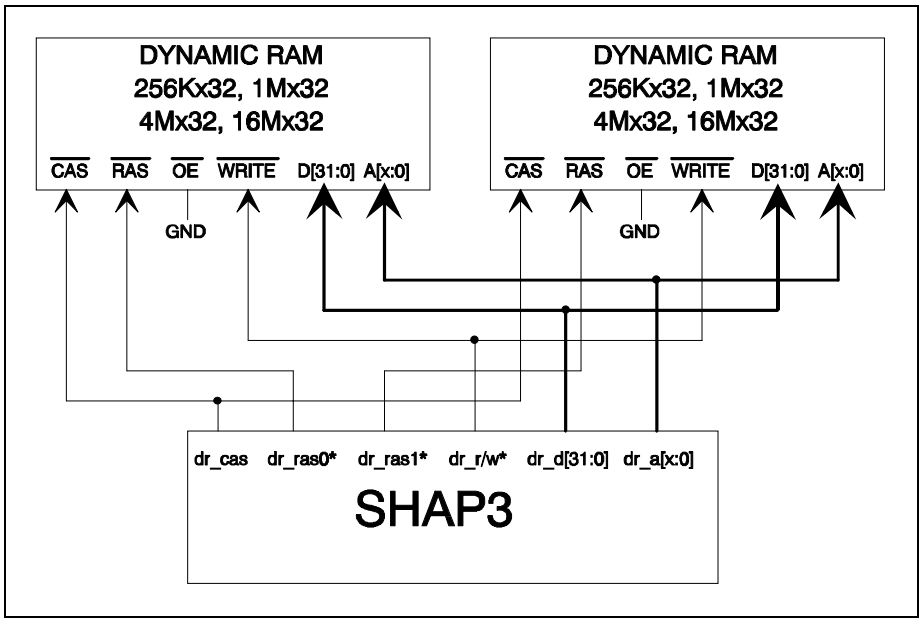


Figure 11-2: Memory configuration with two banks

11.2 Static memory

The SHAP3 can store 32 parameter sets for each class. If this feature is used an external static memory of 16Kx8 bits, 25 ns access time must be connected to the SHAP3. Figure 11-3 shows one possibility by using a 15 ns, 32Kx8 cache RAM.

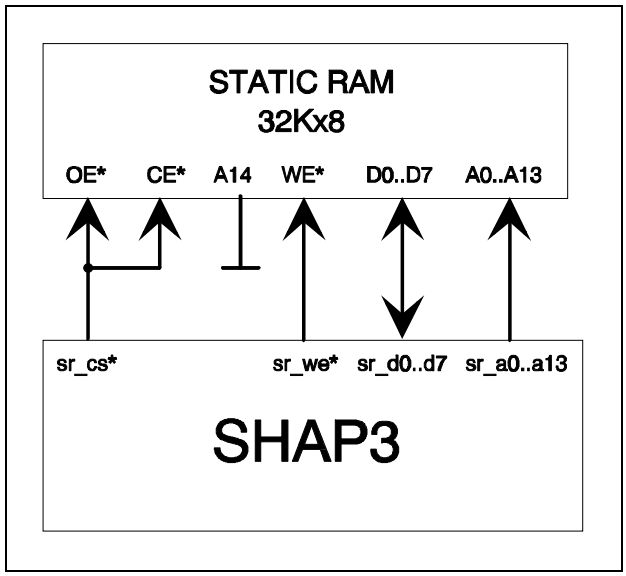


Figure 11-3: Static RAM connected to SHAP3

12 Register summary

The SHAP3 has a number of dedicated registers who's task and function do not alter. Besides the dedicated function twenty of *all purpose registers* are used to temporary store data which must be read from or written to the various memories. The table below gives the addresses for all registers present in the SHAP3.

address	Register
0..3	Header mask register 0..3
4..7	Leaky bucket parameter select register 0..3
8	Input control register
9	Interrupt control register
10	Dynamic RAM control register
11	Class select register
12	Leaky bucket control register
13	Parameter set control register
14..33	All purpose data register 0..19
34	Transfer control register
35	Input status register
36	Cell out control register

12.1 Dedicated registers

Header mask register (HDM) **[Address 0..3]**

Reg. No.	Function
0	header mask of the cell octet 1
1	header mask of the cell octet 2
2	header mask of the cell octet 3
3	header mask of the cell octet 4

Leaky bucket parameter select register (CSL) **[Address 4..7]**

The *Leaky Bucket parameter select registers* are used when operating in single Leaky Bucket mode. Each bit selects for a class if parameter 1 or parameter 2 is active.

- If the bit is cleared parameter 1 is active
- If the bit is set parameter 2 is active

Leaky bucket parameter select register 0 controls class 0..7, register one class 8..15, etc. The LS bit controls the lowest class. Thus bit 0 of *Leaky Bucket parameter select register* 0 controls class 0, bit 1 controls class 1, etc.

MS							**LS**	
Par. sel Class 7	**Par. sel. Class 6**	**Par. sel. Class 5**	**Par. sel. Class 4**	**Par. sel. Class 3**	**Par. sel. Class 2**	**Par. sel. Class 1**	**Par. sel. Class 0**	**REG 0**
Par. sel. Class 15	**Par. sel. Class 14**	**Par. sel. Class 13**	**Par. sel. Class 12**	**Par. sel. Class 11**	**Par. sel. Class 10**	**Par. sel. Class 9**	**Par. sel. Class 8**	**REG 1**
Par. sel. Class 23	**Par. sel. Class 22**	**Par. sel. Class 21**	**Par. sel. Class 20**	**Par. sel. Class 19**	**Par. sel. Class 18**	**Par. sel. Class 17**	**Par. sel. Class 16**	**REG 2**
Par. sel. Class 31	**Par. sel. Class 30**	**Par. sel. Class 29**	**Par. sel. Class 28**	**Par. sel. Class 27**	**Par. sel. Class 26**	**Par. sel. Class 25**	**Par. sel. Class 24**	**REG 3**

Input control register (IOC) [Address 8]

Bit	Function if bit cleared		Function if bit set
0	Link input disabled		Link input enabled
1	Input HEC byte ignored		Input HEC check on
2	Input idle cells passed on		Input idle cells removed
3	Assert RxEnb* at the end of the cell		Assert RxEnb* directly
4	Time stamp counter increment each RxClk cycle		Time stamp counter increment every 53 or 56 RxClk cycles (depends on the input cell type)
[6:5]			
	value	Input cell type	Dram write format
	00	53 bytes	53 bytes + time stamp
	01	53 bytes + 3 bytes tag	53 bytes,+ time stamp (conversion 56 to 53 bytes)
	10	56 bytes	56 bytes
	11	53 bytes + 3 bytes tag	53 bytes + 3 bytes tag
7	Assert RxEnb* with TxClav		Assert RxEnb* with TxClav and FIFO full

Interrupt control register (IC) [Address 9]

Bit(s)	Function if bit cleared	Function if bit set
0	disable early RxSOC interrupt	enable early RxSOC interrupt
1	disable late RxSOC interrupt	enable late RxSOC interrupt
2	disable missing RxClk interrupt	enable missing RxClk interrupt
3	disable counter overflow interrupt of the queue control unit	enable counter overflow interrupt of the queue control unit
4	disable FIFO contains more than 2 cells interrupt	enable FIFO contains more than 2 cells interrupt
5	disable cell discard interrupt	enable cell discard interrupt
[7:6]	Reserved for future extension	

Dynamic RAM control register (DRC) [Address 10]

Bit(s)	Function if bit cleared	Function if bit set
[1:0]	These bits determine the type and size of the external memory. For more details see the table below	
[4:2]	value / refresh cycle	
	000 one refresh every 53(or 56) clk cycles (depending on bit 1 of COLB) 001 one refresh every 2*53(or 56) clk cycles (depending on bit 1 of COLB) 010 one refresh every 3*53(or 56) clk cycles (depending on bit 1 of COLB) 011 one refresh every 4*53(or 56) clk cycles (depending on bit 1 of COLB) 100 one refresh every 5*53(or 56) clk cycles (depending on bit 1 of COLB) 101 one refresh every 6*53(or 56) clk cycles (depending on bit 1 of COLB) 110 one refresh every 7*53(or 56) clk cycles (depending on bit 1 of COLB) 111 one refresh every 8*53(or 56) clk cycles (depending on bit 1 of COLB)	
5	Additional CAS delay	No additional CAS delay
6	must be cleared	
7	CLP level is not used	Discard incoming cells with CLP=1 if level> CLP-level

The SHAP3 is intelligent that it uses only the lower address bits if smaller memory is programmed. The table below shows how the various memory bits map on the output port of the SHAP3.

Bits 1 and 0 of the Dynamic RAM control register:

Bits [1:0]	Address lines used*	RAM type	Address range bank 0 (hex)	Address range bank 1 (hex)	Cell storage 1 bank	Cell storage 2 banks
00	0 to 8	256kx32	0 .. 3FFFF	40000 .. 7FFFF	17.280	35.712
01	0 to 9	1Mx32	0 .. FFFFF	100000 .. 1FFFFF	72.576	146.304
10	0 to 10	4Mx32	0 .. 3FFFFF	400000 .. 7FFFFF	293.760	589.248
11	0 to 11	16Mx32	0 .. FFFFFF	1000000 .. 1FFFFFF	1.178.496	2.358.720

*These address lines should be connected to the DRAM address line. These address lines are used twice. Once during a RAS and once during a CAS access. One extra address line is used internally to select between bank 0 and bank 1.

Class select register (CLA) [Address 11]

Bit(s)	Function
[4:0]	class select register
5	Increment parameter pointer
6	Decrement parameter pointer
7	Reserved for future extension

The increment decrement bits work only if the parameter selection is under software control. (See bit 4 of *the cell out control register*).

Leaky bucket control register (LBC) [Address 12]

Bit(s)	Function if bit cleared	Function if bit set
[4:0]	priority decoder static and round robin select	
	value	class priorities
	00000	no classes static / all classes round robin
	00001	classes 0-1 static / classes 2-31 round robin
	00010	classes 0-3 static / classes 4-31 round robin
	00011	classes 0-5 static / classes 6-31 round robin
	00100	classes 0-7 static / classes 8-31 round robin
	00101	classes 0-9 static / classes 10-31 round robin
	00110	classes 0-11 static / classes 12-31 round robin
	00111	classes 0-13 static / classes 14-31 round robin
	01000	classes 0-15 static / classes 16-31 round robin
	01001	classes 0-17 static / classes 18-31 round robin
	01010	classes 0-19 static / classes 20-31 round robin
	01011	classes 0-21 static / classes 22-31 round robin
	01100	classes 0-23 static / classes 24-31 round robin
	01101	classes 0-25 static / classes 26-31 round robin
	01110	classes 0-27 static / classes 28-31 round robin
	01111	classes 0-29 static / classes 30-31 round robin
	10000	all classes static / no class round robin
5	disable Leaky Bucket operations	enable Leaky Bucket operations
6	must be cleared	
7	Dual Leaky Bucket mode	Single Leaky Bucket mode

Parameter set control register (PSC) [Address 13]

Bit(s)	Function if bit cleared	Function if bit set
[4:0]	Parameter set	
[6:5]		
	value	decrement mode
	00	new parameter set pointer = old parameter set pointer - 1
	01	new parameter set pointer = old parameter set pointer * ¾
	10	new parameter set pointer = old parameter set pointer * ½
	11	new parameter set pointer = old parameter set pointer * ¼ (See also the conversion table in chapter 8)
7	write parameter set pointer to external static RAM	write parameter set to external static RAM

Transfer control register (TRC) [Address 34]

Bit(s)	Function if bit cleared	Function if bit set
0	Transfer from device to up register	Transfer from up register to device
[3:1]		
	value	action:
	000	read/write from register to the class lookup table in the external dynamic RAM
	001	read/write from register to the on chip queue control RAM
	010	read/write from register to the on chip Leaky Bucket Level/Parameter RAM (depending on bit 4 of the transfer control register)
	011	read/write from register to the parameter/pointer (depending on bit 7 of the parameter set control register) of the external static RAM
	100	read/write from register to the cell interface unit
4	read/write Parameter	read/write Level
5	toggles every 53/56 clk cycles depending on bit 1 of the cell out control register	
6	cell interface busy	cell interface ready
7	transfer still pending	transfer is ready

Input status register (IS) [Address 35]

Bit(s)	Alarm condition
0	Early cell sync
1	Late cell sync
2	Missing RxClk
3	Counter overflow
4	FIFO contains more than 2 cells
5	Cell discarded due to FIFO full
[7:6]	don't care

Cell out control register (COLB) [Address 36]

Bit(s)	Function if bit cleared	Function if bit set
0	cell out ctrl produces gaps of one cell	cell out ctrl produces idle cells
1	cell out ctrl send 53 bytes/cell	cell out ctrl send 56 bytes/cell
2	Idle cell with 3 zero +tag bytes if bit 1 is set	Idle cell with 51 byte idle payload if bit 1 is set.
3	must be cleared	
4	parameter set hardware controlled inc/dec	parameter set software contr. inc/dec
[6:5]		
	value action	
	00 All Classes work normally 01 Class 0 works in circular mode, all others normal 10 Class 8..23 in circular mode, Class 0..7 and 24..31 in normal mode 11 All Classes in circular mode	
7	must be cleared	

If cell size is 53 bytes (bit 1 is clear) bit 2 is don't care.

12.2 All purpose data registers

The device has 20 *all purpose data register* used for data transfer between the processor and the various storage locations. The following storage locations can be identified:

- Class lookup table in external dynamic RAM
- Cell storage in external dynamic RAM
- Parameter set table in external static RAM
- Parameter set pointer table in external static RAM
- Queue control parameters in internal RAM
- Leaky bucket parameters in internal RAM
- Leaky bucket level in internal RAM

The *transfer control register* described above specifies:

- Which storage location is accessed
- If a transfer is made from or to a storage location

For each of the seven possible storage locations a description of the *all purpose data registers* is given.

Transferring from/to class lookup table

Register	Bits	Function
0	[4:0]	selected class for extracted cell header address + 0
	[7:5]	don't care
1	[4:0]	selected class for extracted cell header address + 1
	[7:5]	don't care
2	[4:0]	selected class for extracted cell header address + 2
	[7:5]	don't care
3	[4:0]	selected class for extracted cell header address + 3
	[7:5]	don't care
4	[7:0]	Memory address bits [7:0]

5	[X:0] [7:X+1]	Memory address bits [X:8] Should be zero
6	[7:0]	Should be zero
7	0 [7:1]	Should be zero don't care
8..19	[7:0]	don't care

The register description given above is valid if reading data from the look-up table or writing data to the look-up table. The value of X equals the number of header extraction mask bits set minus 2. However the command to access the look-up table can be used in general to read from or write to the dynamic RAM. Although the complete 32 bits are transferred the look-up function of the SHAP3 uses only the LS 5 bits of each byte.

The format shown below is valid for a general memory access.

Transferring from/to memory using class lookup table transfer command:

Register	Bits	Function
0	[7:0]	Byte from memory bits [7:0]
1	[7:0]	Byte from memory bits [15:8]
2	[7:0]	Byte from memory bits [23:16]
3	[7:0]	Byte from memory bits [31:24]
4	[7:0]	Memory address bits [7:0]
5	[5:0]	Memory address bits [15:8]
6	[7:0]	Memory address bits [23:16]
7	0	Memory address bit 24
	[7:1]	don't care
8..19	[7:0]	don't care

How the memory bits map onto the physical ram bits depends on the size of the memory chosen (Bits 0&1 of the *Dynamic RAM Control register*).

Transferring from or to on chip Leaky Bucket Level RAM

Register	Bits	Function
0	0	don't care
	[7:1]	Leaky Bucket 1 level fraction bits [6:0]
1	[7:0]	Leaky Bucket 1 level bits [7:0]
2	[7:0]	Leaky Bucket 1 level bits [15:8]
3	[7:0]	Leaky Bucket 1 level bits [23:16]
4	0	don't care
	[7:1]	Leaky Bucket 2 level fraction bits [6:0]
5	[7:0]	Leaky Bucket 2 level bits [7:0]
6	[7:0]	Leaky Bucket 2 level bits [15:8]
7	[7:0]	Leaky Bucket 2 level bits [23:16]
8..19	[7:0]	don't care

The level of the LB's for each class can be set independent of the rest of the Leaky Bucket parameters.

Transferring from/to Leaky Bucket Parameter set in internal or external RAM

Register	Bits	Function
0	[7:0]	Leaky Bucket 1 limit bits [7:0]
1	[7:0]	Leaky Bucket 1 limit bits [15:8]
2	[7:0]	Leaky Bucket 1 limit bits [23:16]
3	[7:0]	Leaky Bucket 2 limit bits [7:0]
4	[7:0]	Leaky Bucket 2 limit bits [15:8]
5	[7:0]	Leaky Bucket 2 limit bits [23:16]
6	[7:0]	Leaky Bucket 1 splash bits [7:0]
7	[7:0]	Leaky Bucket 2 splash bits [7:0]
8	[3:0]	Leaky Bucket 1 leak bits [3:0]
	[7:4]	Leaky Bucket 2 leak bits [3:0]
9	[5:0]	parameter set pointer (when reading)
	[7:6]	don't care
10..19	[7:0]	don't care

The parameter set pointer is related with the static parameter RAM. It points to the parameter set currently in use. As this value can be controlled externally (using the PAR_INC and PAR_DEC signals) it is always possible that the actual parameter set pointer differs from the one read.

Transferring from/to Leaky Bucket Parameter set pointer table in external RAM

Register	Bits	Function
0..8	[7:0]	don't care
9	[5:0]	parameter set pointer
	[7:6]	don't care
10..19	[7:0]	don't care

The parameter set pointer is special. When reading a parameter set from the external static RAM the parameter set as well as the pointer are loaded. However when writing to the external static RAM first the pointer must be written and then a second write command is needed to transfer the parameter set to the entry to which the pointer is pointing.

Transferring from/to on chip queue control RAM

Register	Bits	Function if bit is cleared	Function if bit is set
0	[7:0]	queue base address bits [7:0]	
1	[4:0]	queue base address bits [12:8]	
	5	if the counter reaches the maximum value, the counter holds this value.	If the counter reaches the maximum value, it rolls over and sets the status bit.
	6	count all discarded cells (discarded because the queue is full or level>CLP level).	count only cells discarded because the queue is full.
	7	This bit is used by the queue control algorithm and must be cleared when the queue control parameters are initialised.	
2	[7:0]	queue size in cells bits [7:0]	
3	[7:0]	queue size in cells bits [15:8]	
4	[4:0]	queue size in cells bits [20:16]	
	[7:5]	don't care	
5	[7:0]	CLP level bits [7:0]	
6	[7:0]	CLP level bits [15:8]	
7	[4:0]	CLP level bits [20:16]	
	[7:5]	don't care	
8	[7:0]	discarded cells counter bits [7:0]	
9	[7:0]	discarded cells counter bits [15:8]	
10	[3:0]	discarded cells counter bits [19:16]	
	[7:4]	don't care	
11	[7:0]	Write pointer bits [7:0].[6]	
12	[7:0]	Write pointer bits [15:8].	

[6]The Write pointer and Read pointer bits must be cleared when the queue control parameters are initialised.

Register	Bits	Function if bit is cleared	Function if bit is set
13	[4:0]	Write pointer bits [20:16].	
	[7:5]	don't care	
14	[7:0]	Read pointer bits [7:0].	
15	[7:0]	Read pointer bits [15:8].	
16	[4:0]	Read pointer bits [20:16].	
	[7:5]	don't care	
17	[7:0]	queue level in cells bits [7:0]. This register is automatically updated when a parameter is read from on chip queue control RAM.	
18	[7:0]	queue level in cells bits [15:8] This register is automatically updated when a parameter is read from on chip queue control RAM.	
19	[4:0]	queue level in cells bits [20:16].	
	5	queue is not empty	queue is empty
	6	queue is not full	queue is full
	7	queue length is <= than the CLP level	queue length is above the CLP level

All purpose data register 19 is automatically updated when the queue parameters are read from the on chip queue control RAM.

Transferring from/to the cell interface unit

Register	Function
0	cell data 0
1	cell data 1
2	cell data 2
3	cell data 3
19 .. 4	don't care

The cell interface unit allows the user to access a cell in the memory reading or writing 4 bytes at the time.

13 Signal description

13.1 ATM input port

Name	I/O	Description
RxData[7:0]	Input	**ATM in data:** These signals hold the 8 bit wide data. All arriving data between the last byte of the previous cell and the first byte of the following cell (indicated by the SOC signal) is ignored.
RxSOC	Input	**ATM in cell sync:** This signal is high during the first byte of an arriving cell. After this signal is high the following 52 (or 55) bytes should contain valid data. The SHAP3 waits for another RxSOC signal after reading a complete cell.
RxClk	Input	**ATM in byte clock:** This signal is the clock of the incoming data. Data is sampled after the rising edge of this signal.
RxEnb*	Output	**Enable:** This signal is asserted by the SHAP3 to indicate that RxData and RxSOC will be sampled at the end of the next cycle

13.2 ATM output port

The signal TxClk (ATM out byte clock) is the system clock clk. This signal is the clock of the outgoing data. Data of the ATM output ports changes after the rising edge of this signal.

Name	I/O	Description
TxData[7:0]	Output	**ATM data output:** These signals output the 8 bit wide data.
TxSOC	Output	**ATM cell sync output:** This signal is high during the first byte of a cell.
TxOE*	Input	**ATM cell sync, data enable:** If low it enables the cell sync and the data signal. If high the cell sync and the data signal is tri-state.
TxEnb*	Output	**Enable:** It will be low when a cell (including idle cells) is being transmitted at the output.
TxClav	Input	**Cell Available:** This signal is an active high signal from an external device, The external device should assert this signal to indicate it can accept the transfer of a complete cell.

13.3 Dynamic memory interface

Name	I/O	Description
dr_d[31:0]	I/O	**Dynamic memory data bus:** These signals hold data to be exchanged between the SHAP3 and the external dynamic memory.
dr_a[11:0]	Output	**Dynamic memory address bus:** These signals are used to select an entry in the external dynamic memory.
dr_r/w*	Output	**Dynamic memory read/not write:** If *low* data is written from the SHAP3 to the memory. If *high* data is read from the memory to the SHAP3.
dr_ras0*	Output	**Dynamic Memory Row Address Strobe bank 0:** It is used to signal that the address bus holds a valid row address for memory bank 0.
dr_ras1*	Output	**Dynamic Memory Row Address Strobe bank 1:** It is used to signal that the address bus holds a valid row address for memory bank 1.
dr_cas*	Output	**Dynamic Memory Column Address Strobe:** It is used to signal that the address bus holds a valid column address.

13.4 Static memory interface

The static memory is optional. It can store up to 32 parameter sets for each Leaky Bucket and for each class.

Name	I/O	Description
sr_d[7:0]	I/O	**Static memory data bus:** They hold data to be exchanged between the SHAP3 and the external static memory.
sr_a[13:0]	Output	**Static memory address bus:** They are used to select an entry in the external static memory.
sr_we*	Output	**Static memory read/not write:** If *low* data is written from the SHAP3 to the memory. If *high* data is read from the memory to the SHAP3.
sr_cs*	Output	**Static memory chip select:** If this signal is *low* the transfer to/from static RAM starts.

13.5 Microprocessor interface

Name	I/O	Description
up_d[7:0]	I/O	**Microprocessor data bus:** This bus holds data to be exchanged between the SHAP3 and the microprocessor.
up_a[5:0]	Output	**Microprocessor address bus:** These signals ared used to select one of the internal registers of the SHAP3.
up_rw*	Output	**Microprocessor read/not write:** If *low* data is written from the microprocessor to the SHAP3. If *high* data is read from the SHAP3.
up_cs*	Output	**Microprocessor chip select:** If this signal is *high* the SHAP3 ignores all other signal on it's microprocessor bus. If this signal is *low* the SHAP3 accepts the signals on it's microprocessor bus. When up_r/w* is low this signal should be glitch-free as in that case data is loaded into the SHAP3 at it's rising edge.
up_irq*	Output	**Microprocessor Interrupt Request**. This is an output signal. If this signal is *low* the SHAP3 signals to the processor that an interrupt condition is pending inside the SHAP3. Otherwise no interrupt is pending inside the SHAP3.
DMAREQ*	Output	**DMA Request**. This signal is internally connected to bit 7 of the transfer control register. If this signal is high the DMA controller can transfer data to/from the SHAP3. If this signal is low the DMA controller should stop sending/reading data from/to the SHAP3.

13.6 Parameter select

Name	I/O	Description
class[4:0]	Input	**Selected class number**. These signals hold the class number for the increment or decrement operation started by a Par_inc or Par_dec puls.
Par_dec	Input	**Parameter set decrement**. A low to high pulse decrements the parameter set pointer of the class given by the class signal and transfers the calculated parameter set (of this class) to the on chip parameter RAM.
Par_inc	Input	**Parameter set increment**. A low to high pulse increments the parameter set pointer of the class given by the class signal and transfers the calculated parameter set (of this class) to the on chip parameter RAM.

13.7 System signals

Name	I/O	Description
clk	Input	**System clock**. This is an input signal. In the SHAP3 this clock is used for the memory controller, Leaky Bucket algorithm and the generation of the output signals.
Reset*	Input	**System reset**. This is an active low input signal. This signal will cause the device to enter the initial state.
test*	Input	**Signal used for scan test**. This signal should be high for normal operation. The signal is internally connected to a 9 KOhm Pull-up. So this pin can be left unconnected.
Tdi	Input	**JTAG Test Data Input**. The signal is internally connected to a 9 KOhm Pull-up
tms	Input	**JTAG Test Mode Select Input**. The signal is internally connected to a 9 KOhm Pull-up
tck	Input	**JTAG Test Clock**. Connect this pin to ground for normal operation.
Trst*	Input	**JTAG Test Reset**. The signal is internally connected to a 9 KOhm Pull-up. Connect this pin to ground for normal operation.
Tdo	Output	**JTAG Test Data Output**

14 Pin assignment

Package: Plastic quad flat pack 160

Pin	Signal	I/O	Description
1	up_d4	in/out	microprocessor data bus bit 4
2	up_d5	in/out	microprocessor data bus bit 5
3	up_d6	in/out	microprocessor data bus bit 6
4	up_d7	in/out	microprocessor data bus bit 7
5	gnd_pad	ground	0 Volt pad ground
6	vdd_pad	power	5 Volt pad power
7	up_a0	input	microprocessor addr. bus bit 0
8	up_a1	input	microprocessor addr. bus bit 1
9	up_a2	input	microprocessor addr. bus bit 2
10	up_a3	input	microprocessor addr. bus bit 3
11	up_a4	input	microprocessor addr. bus bit 4
12	up_a5	input	microprocessor addr. bus bit 5
13	test*	input	test signal used for scan test
14	clk	input	system clock
15	gnd_clk	ground	0 Volt
16	vdd_clk	power	5 Volt
17	trst*	input	JTAG test reset input
18	tdo	output	JTAG test data output
19	tck	input	JTAG test clock input
20	tms	input	JTAG test mode select input
21	tdi	input	JTAG test data input
22	sr_d7	in/out	static RAM data bus bit 7
23	sr_d6	in/out	static RAM data bus bit 6
24	sr_d5	in/out	static RAM data bus bit 5
25	gnd_pad	ground	0 Volt
26	vdd_pad	power	5 Volt
27	sr_d4	in/out	static RAM data bus bit 4
28	sr_d3	in/out	static RAM data bus bit 3
29	sr_d2	in/out	static RAM data bus bit 2
30	sr_d1	in/out	static RAM data bus bit 1
31	sr_d0	in/out	static RAM data bus bit 0
32	sr_a13	output	static RAM addr. bus bit 13
33	sr_a12	output	static RAM addr. bus bit 12
34	sr_a11	output	static RAM addr. bus bit 11
35	gnd_core	ground	0 Volt
36	vdd_core	power	5 Volt
37	sr_a10	output	static RAM addr. bus bit 10

Pin	Signal	I/O	Description
38	sr_a9	output	static RAM addr. bus bit 9
39	sr_a8	output	static RAM addr. bus bit 8
40	sr_a7	output	static RAM addr. bus bit 7
41	sr_a6	output	static RAM addr. bus bit 6
42	sr_a5	output	static RAM addr. bus bit 5
43	sr_a4	output	static RAM addr. bus bit 4
44	sr_a3	output	static RAM addr. bus bit 3
45	gnd_pad	ground	0 Volt
46	vdd_pad	power	5 Volt
47	sr_a2	output	static RAM addr. bus bit 2
48	sr_a1	output	static RAM addr. bus bit 1
49	sr_a0	output	static RAM addr. bus bit 0
50	sr_we*	output	static RAM write/not read
51	sr_cs*	output	static RAM chip select
52	class4	input	class value bit 4
53	class3	input	class value bit 3
54	class2	input	class value bit 2
55	gnd_core	ground	0 Volt
56	vdd_core	power	5 Volt
57	class1	input	class value bit 1
58	class0	input	class value bit 0
59	Par_dec	input	decrement parameter set
60	Par_inc	input	increment parameter set
61	TxEnb*	output	ATM output port low when cell is being transmitted
62	TxClav	input	ATM output port cause the SHAP3 stop sending cells
63	TxSOC	output	ATM output port cell sync
64	TxOE*	input	ATM output port tristate data and cell sync
65	gnd_pad	ground	0 Volt
66	vdd_pad	power	5 Volt
67	TxData7	output	ATM output port data bit 7
68	TxData6	output	ATM output port data bit 6
69	TxData5	output	ATM output port data bit 5
70	TxData4	output	ATM output port data bit 4
71	TxData3	output	ATM output port data bit 3
72	TxData2	output	ATM output port data bit 2
73	TxData1	output	ATM output port data bit 1
74	TxData0	output	ATM output port data bit 0
75	gnd_pad	ground	0 Volt
76	vdd_pad	power	5 Volt
77	dr_d31	in/out	dynamic RAM data bus bit 31
78	dr_d30	in/out	dynamic RAM data bus bit 30
79	dr_d29	in/out	dynamic RAM data bus bit 29
80	dr_d28	in/out	dynamic RAM data bus bit 28
81	dr_d27	in/out	dynamic RAM data bus bit 27
82	dr_d26	in/out	dynamic RAM data bus bit 26
83	dr_d25	in/out	dynamic RAM data bus bit 25
84	dr_d24	in/out	dynamic RAM data bus bit 24
85	dr_d23	in/out	dynamic RAM data bus bit 23
86	gnd_pad	ground	0 Volt
87	vdd_pad	power	5 Volt
88	dr_d22	in/out	dynamic RAM data bus bit 22
89	dr_d21	in/out	dynamic RAM data bus bit 21
90	dr_d20	in/out	dynamic RAM data bus bit 20
91	dr_d19	in/out	dynamic RAM data bus bit 19

Pin	Signal	I/O	Description
92	dr_d18	in/out	dynamic RAM data bus bit 18
93	dr_d17	in/out	dynamic RAM data bus bit 17
94	dr_d16	in/out	dynamic RAM data bus bit 16
95	dr_r/w*	output	dynamic RAM read/not write
96	dr_ras1*	output	dynamic RAM row address strobe bank 1
97	dr_ras0*	output	dynamic RAM row address strobe bank 0
98	gnd_pad	ground	0 Volt
99	vdd_pas	power	5 Volt
100	gnd_core	ground	0 Volt
101	vdd_core	power	5 Volt
102	dr_cas*	output	dynamic RAM column address strobe
103	dr_a11	output	dynamic RAM addr. bus bit 11
104	dr_a10	output	dynamic RAM addr. bus bit 10
105	dr_a9	output	dynamic RAM addr. bus bit 9
106	dr_a8	output	dynamic RAM addr. bus bit 8
107	dr_d15	in/out	dynamic RAM data bus bit 15
108	dr_d14	in/out	dynamic RAM data bus bit 14
109	dr_d13	in/out	dynamic RAM data bus bit 13
110	dr_d12	in/out	dynamic RAM data bus bit 12
111	dr_d11	in/out	dynamic RAM data bus bit 11
112	gnd_pad	ground	0 Volt
113	vdd_pad	power	5 Volt
114	dr_d10	in/out	dynamic RAM data bus bit 10
115	dr_d9	in/out	dynamic RAM data bus bit 9
116	dr_d8	in/out	dynamic RAM data bus bit 8
117	dr_a7	output	dynamic RAM addr. bus bit 7
118	dr_a6	output	dynamic RAM addr. bus bit 6
119	dr_a5	output	dynamic RAM addr. bus bit 5
120	dr_a4	output	dynamic RAM addr. bus bit 4
121	dr_a3	output	dynamic RAM addr. bus bit 3
122	dr_a2	output	dynamic RAM addr. bus bit 2
123	dr_a1	output	dynamic RAM addr. bus bit 1
124	gnd_pad	ground	0 Volt
125	vdd_pad	power	5 Volt
126	dr_a0	output	dynamic RAM addr. bus bit 0
127	dr_d7	in/out	dynamic RAM data bus bit 7
128	dr_d6	in/out	dynamic RAM data bus bit 6
129	dr_d5	in/out	dynamic RAM data bus bit 5
130	dr_d4	in/out	dynamic RAM data bus bit 4
131	dr_d3	in/out	dynamic RAM data bus bit 3
132	dr_d2	in/out	dynamic RAM data bus bit 2
133	dr_d1	in/out	dynamic RAM data bus bit 1
134	dr_d0	in/out	dynamic RAM data bus bit 0
135	gnd_pad	ground	0 Volt
136	vdd_pad	power	5 Volt
137	RxData0	input	ATM input port data bit 0
138	RxData1	input	ATM input port data bit 1
139	RxData2	input	ATM input port data bit 2
140	RxData3	input	ATM input port data bit 3
141	RxData4	input	ATM input port data bit 4
142	RxData5	input	ATM input port data bit 5
143	RxData6	input	ATM input port data bit 6
144	RxData7	input	ATM input port data bit 7
145	gnd_core	ground	0 Volt
146	vdd_core	power	5 Volt
147	RxSOC	input	ATM input port cell sync

Pin	Signal	I/O	Description
148	RxEnb*	output	ATM input port device sending data should stop sending data
149	RxClk	input	ATM input port byte clock
150	Reset*	input	system reset
151	DMAREQ*	output	SHAP3 transfer busy
152	up_irq*	output	microprocessor interrupt req.
153	up_r/w*	input	microprocessor read/not write
154	up_cs*	input	microprocessor chip select
155	gnd_pad	ground	0 Volt
156	vdd_pad	power	5 Volt
157	up_d0	in/out	microprocessor data bus bit 0
158	up_d1	in/out	microprocessor data bus bit 1
159	up_d2	in/out	microprocessor data bus bit 2
160	up_d3	in/out	microprocessor data bus bit 3

All power and ground pads must be connected. Leaving some power or ground pads open while others are connected will cause permanent damage to the device. We have taken a multilayer PCB for our testboard with one GND and one VDD layer. We have used a 47nF ceramic SMD capacitor for each GND, VDD pair. Each device has been decoupled with three Tantal capacitors C of 10 µF. The following figure (Figure 14-1) shows an example of the connection of the power and ground pads. The three power supply nets are decoupled through an external capacity.

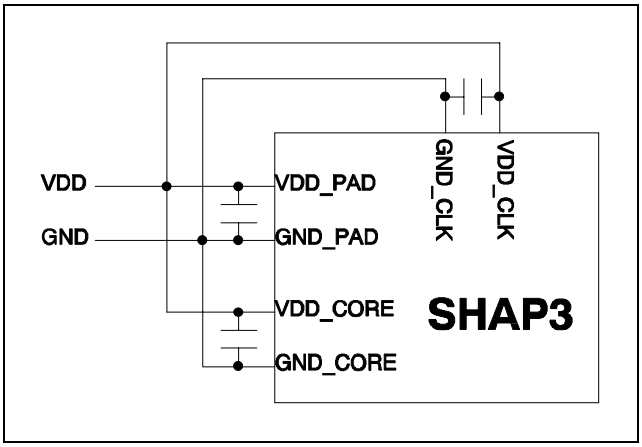


Figure 14-1: Connecting Power and Ground

15 Timing diagrams

The SHAP3 is a full synchronous design except for the microprocessor port and the parameter change port. Therefore there are two groups of timing values: For the microprocessor port and for the rest of the system. The timing for the parameter change port is described before. The timing for the microprocessor port is shown below.

When reading the data is put on the bus immediately after CS* is LOW and read is high. When writing the data is clocked into a pre-load register in the SHAP3 on the rising edge of the CS* signal. After this the data from the pre-load register is transferred to the actual register inside the SHAP3. This requires synchronisation to the system clock of the SHAP3 and can take up to 2 system clock cycles.

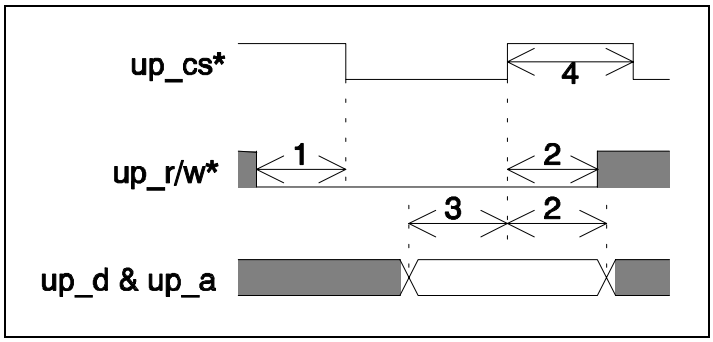


Figure 15-2: Microprocessor write timing

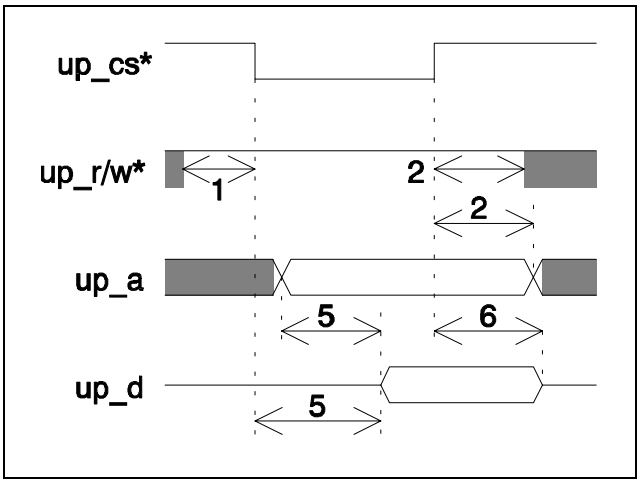


Figure 15-3: Microprocessor read timing

#	Description	Min.	Max.
1	R/W* setup before falling edge of CS*	8 ns	
2	R/W*, data and addr hold after rising edge of CS*	5 ns	
3	Data and addr setup before rising edge of CS*	8 ns	-
4	CS* high after a write access to next CS* low	2 clk cycles[7]	-
5	Data ready after CS* low / addr stable		25 ns
6	Data hold after CS* high		5 ns

For the rest of the system almost all delay times are the same. In fact there are only two delay times:

- Control signals from clock.
  These are marked in the timing diagrams with number **1**.
- Data signal from clock.
  These are marked in the timing diagrams with number **2**.

Note that for the ATM-input signals the setup and hold times are related to the Rxclk, not the system clock.

#	Description	Min.	Max.
1	Control signal delay from system clock	2.8 ns	11.7 ns
2	Data signal delay from system clock	4.0 ns	12.8 ns
3	Signal setup against system clock edge	3.7 ns	-
4	Signal hold after system clock edge	0.9 ns	-

[7]The device must synchronise the data and address written to the internal system clock. For this it is required that at least 2 system clock cycles are present between two write accesses.

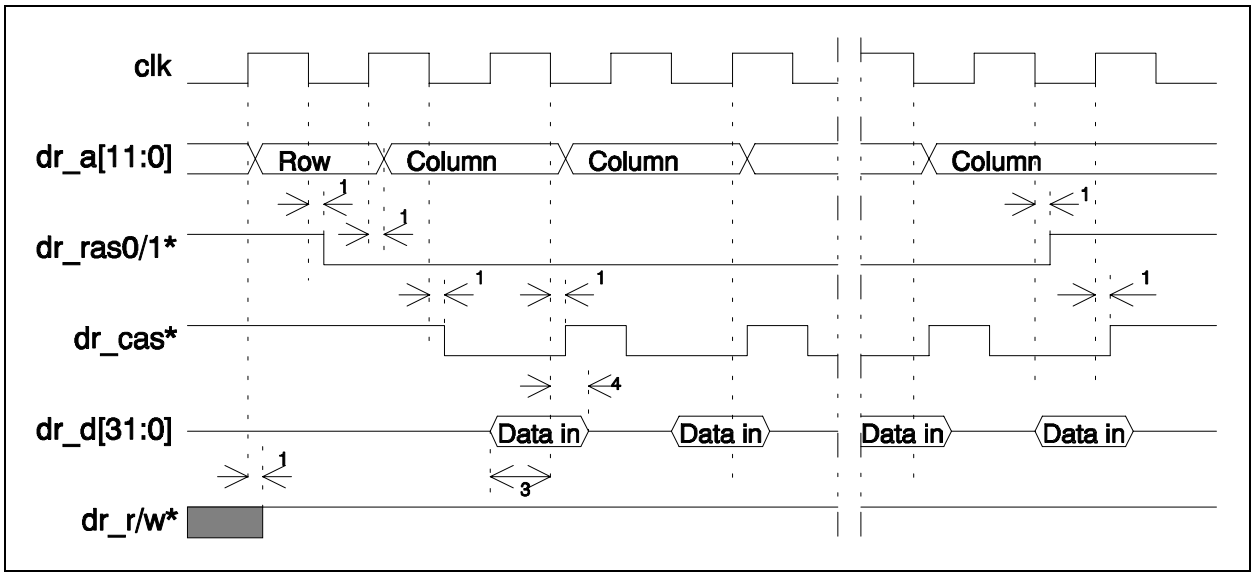


Figure 15-4: Timing diagram read cell of the DRAM

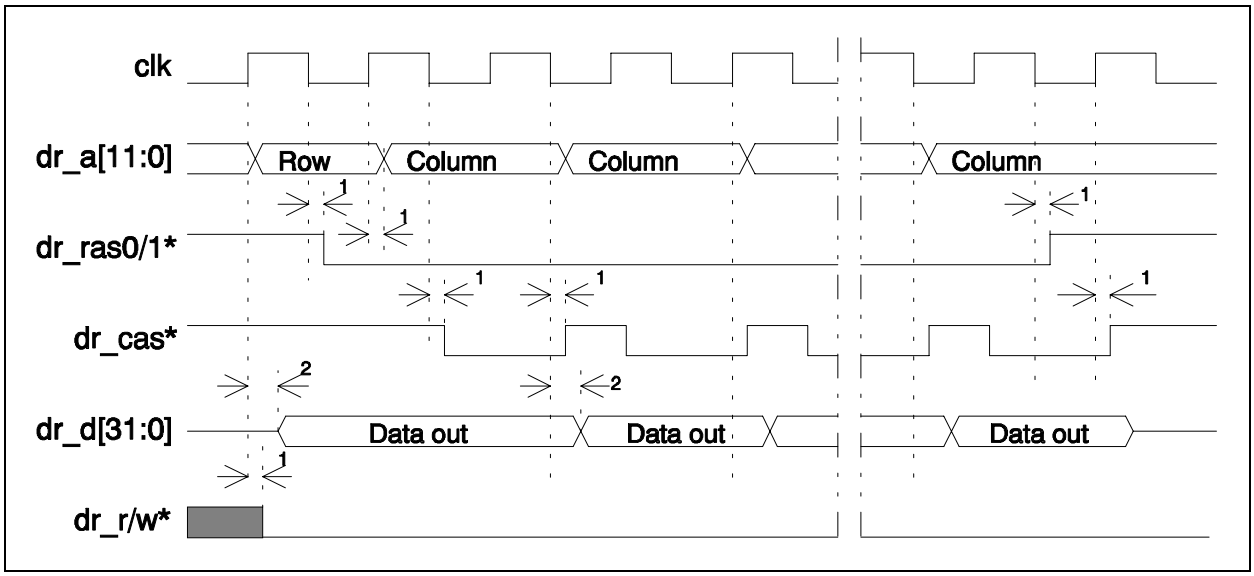


Figure 15-5: Timing diagram write cell of the DRAM

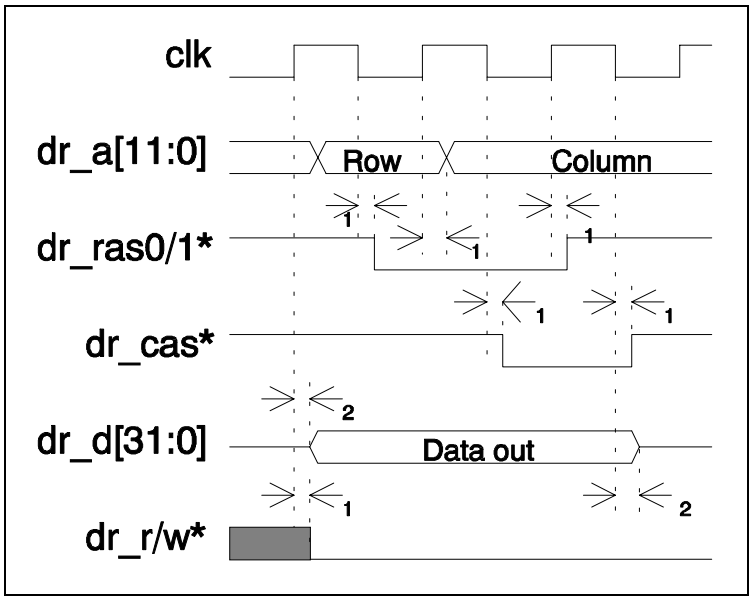


Figure 15-6: Microprocessor DRAM write access

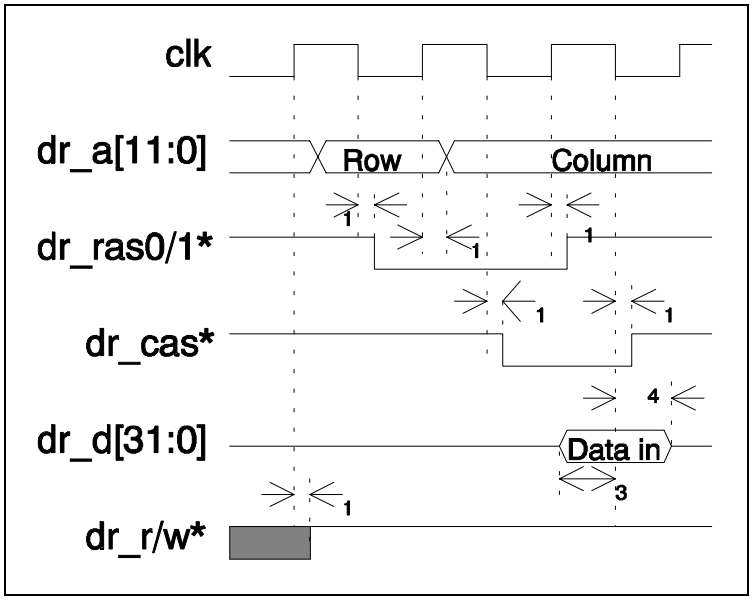


Figure 15-7: Microprocessor DRAM read access

16 Electrical Characteristics

- All outputs are 8 mA CMOS
- All inputs are CMOS
- The input signals test*, tms, tdi and trst are internally connected to a 9 KOhm Pull-up
- Vdd range : 4.5 to 5.5 V

16.1 Absolute Maximum Ratings

Operation of a device outside this range may cause permanent damage to the device and/or affect reliability.

Symbol	Parameter	MIN	MAX	Unit
VDD	DC supply voltage	-0.5	7.0	V
VI	DC input voltage	-0.5	VDD+0.5	V
VO	DC output voltage	-0.5	VDD+0.5	V
TSG	Storage temperature	- 65	+150	°C
TSH	Time of outputs shorted		5	sec
TA	Operating free air temp range	- 40	+ 85	°C

16.2 DC Characteristics

Symbol	Parameter	MIN	MAX	Unit	Conditions
VDD		4.5	5.5	V	
TEMP		- 40	+ 85	°C	
VIL VIH	Low level input voltage High level input voltage	 70%VDD	30%VDD	V V	CMOS inputs and bidirectional
VOL VOH	Low level output voltage High level output voltage	 VDD-0.5	0.5	V V	
IIH IIL	Input leakage, no pullup Input leakage, no pullup	-150 -150	150 150	nA nA	VIN=VDD=5.5V VIN=0 VDD=5.5V

17 Thermal data

The thermal resistance θja (junction - ambient) with a tolerance of ± 15% at still air:

- θja = 38 °C/W

The thermal resistance θjc (junction - case) with a tolerance of ± 15%:

- θjc = 19 °C/W

The thermal resistance θca (case - ambient) with a tolerance of ± 15%:

- θca = 19 °C/W

Operating free air temp range:

- TA = -40 to +85°C

18 Package

18.1 Top view

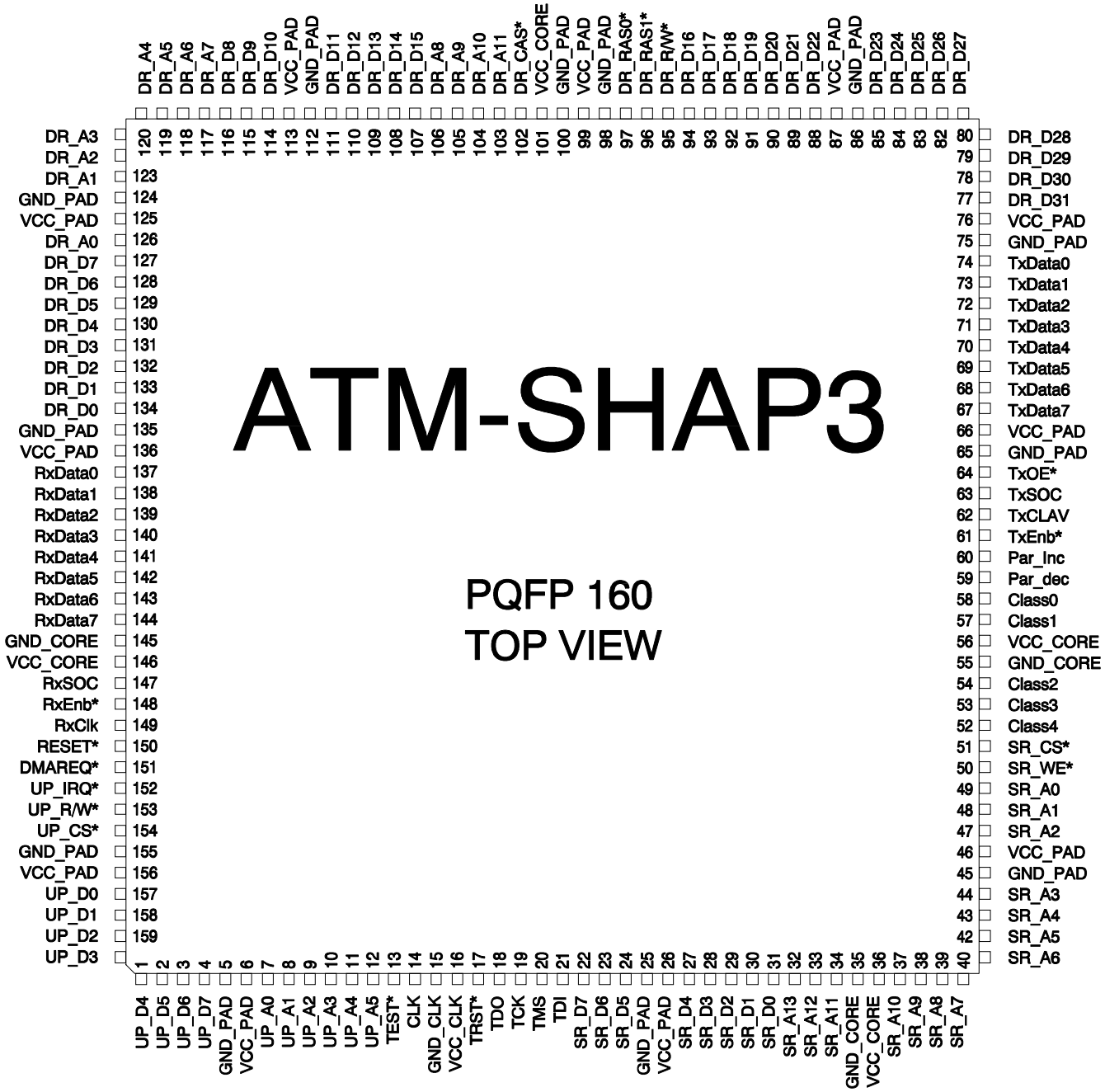


Figure 18-1: SHAP3 package top view

18.2 Mechanical data

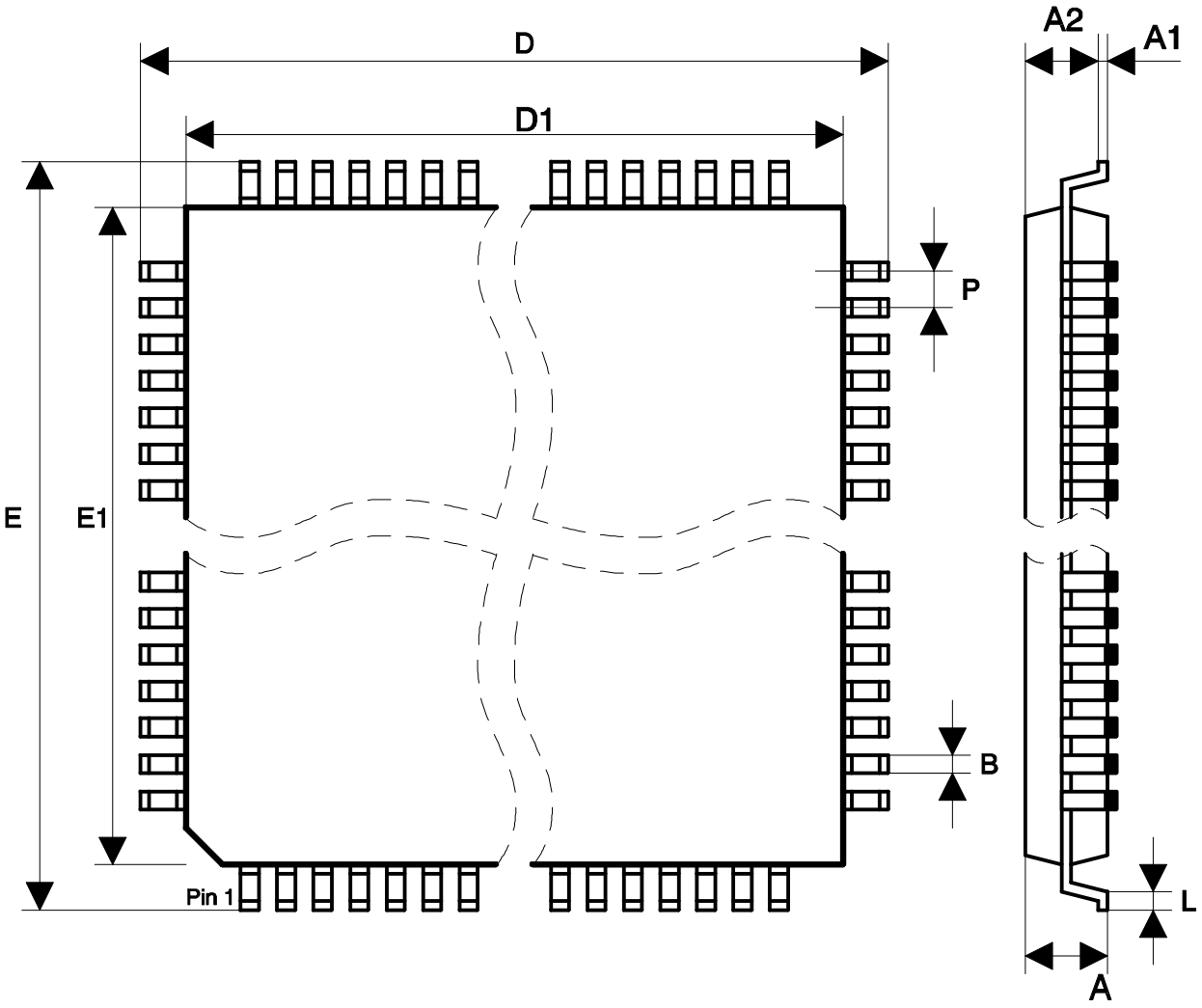


Figure 18-2: Mechanical drawing

160 Pin Copper Leadframe Plastic Quad Flat Pack			
Dimension	Minimum	Nominal	Maximum
A	3.22	3.57	3.97
A1	0.05	0.25	0.5
A2	3.17	3.32	3.47
D	31.65	31.90	32.15
D1	27.90	28.00	28.10
E	31.65	31.90	32.15
E1	27.90	28.00	28.10
L	0.65	0.75	0.95
P		0.65	
B	0.20	0.30	0.40
All dimensions in millimeter			