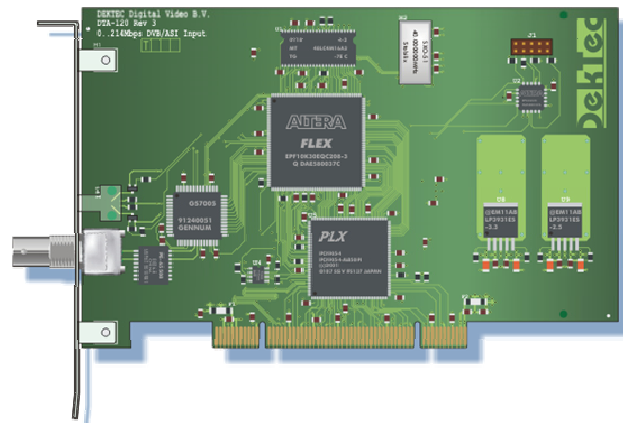
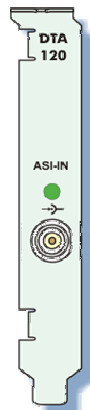


DVB/ASI Input Adapter for PCI Bus

- Accepts Full ASI Range 0...214 Mbit/s
- 8-MBytes Input Buffer
- Time-Stamping Support

FEATURES

- High-speed Transport-Stream input, compliant to DVB/ASI (Asynchronous Serial Interface) as defined in DVB document A010 rev 1 and EN50083
- Supports full DVB/ASI bit-rate range from 0 to 214 Mbit/s
- Adaptive cable equalisation
- Optional time stamp per packet
- Synchronised and raw receive modes
- Counters for bit-rate measurement, 8B/10B code violations and synchronisation errors
- LED indicator shows lock and synchronisation status
- Large jitter tolerance with 8-Mbytes on-board packet buffer
- Automatic recognition and adjustment of inverted DVB/ASI input signals
- Raw-mode option for arbitrary packet size
- On-board PCI Bus master for off-loading host processor
- Scatter/Gather DMA for efficient handling of fragmented host memory
- Optional FEC stripping
- PCI rev 2.2, 32 bit, 33 MHz



APPLICATION AREA

- Universal DVB/ASI input adapter for processing of MPEG-2 Transport Streams

FREE SOFTWARE

- Win-2000/XP and Linux device driver
- DTAPI: Windows/Linux API for developing custom applications
- Command-line recorder with source code
- Windows grabber program (*DtGrabber*)

COMMERCIAL APPLICATIONS

- *StreamXpert™* Transport-Stream Analyser
- *Media Gateway™* DVB-to-IP gateway

KEY ATTRIBUTES

Parameter	Value
Physical Layer	DVB/ASI-C*
DVB/ASI Connector	75 Ω BNC
Input Bit Rate	0...214 Mbit/s
Input Return Loss	17 dB
Error Free Cable Length	300 m
Packet Size in Bytes	188 or 204**
Current Firmware Version	4

* The "-C" suffix indicates Coax as physical-transport medium.

** Arbitrary packet size in raw mode

RELATED PRODUCTS

Type	Description
DTA-100	DVB/ASI Output Adapter for PCI Bus
DTA-122	DVB/SPI Input Adapter for PCI Bus
DTC-320	<i>StreamXpert™</i> TS Analyser Software
asi2ip	<i>Media Gateway™</i> DVB-to-IP gateway
DTC-7X2	Integrated Stream Analyser

Table of Contents

1. General Description	3	3.6. Packet Alignment	12
1.1. Typical Application	3	3.6.1. Fixed Buffer Structure	12
1.2. Software	3	3.6.2. Switching Receive Mode	12
1.3. Block Diagram	4	3.7. Time Stamping	13
1.3.1. Equaliser	4	3.7.1. Block Diagram	13
1.3.2. ASI Receive Logic	4	3.7.2. Time-Stamp Format	13
1.3.3. Receive FIFO	4	4. Configuration Space	14
1.3.4. PCI-9054 Bus Interface	4	5. Target Address Space	16
1.3.5. Register Set	5	5.1. General Registers	19
1.4. References	5	5.1.1. General Control (00h)	19
1.5. Document Overview	5	5.1.2. General Status (04h)	20
2. External Interfaces	6	5.1.3. Programming (08h)	20
2.1. Overview	6	5.1.4. Ref. Clock Count (0Ch)	20
2.2. DVB/ASI Input	6	5.2. Receive Registers	21
2.3. LED Status Indicator	6	5.2.1. Receive Control (24h)	21
2.3.1. LED Power-Up Sequence	7	5.2.2. Receive Status (28h)	23
3. Streaming Data	8	5.2.3. Receive FIFO Load (38h)	25
3.1. DMA vs. Direct Reads	8	5.2.4. Receive Diagnostics (3Ch)	25
3.2. Latencies	8	5.2.5. Receive Loop-Back Data (40h)	26
3.2.1. PCI-Bus Latency	8	5.2.6. Receive Valid Count (50h)	26
3.2.2. Interrupt Latency	9	5.2.7. Receive Violation Count (54h)	26
3.2.3. Scheduling Latency	9	5.2.8. Receive-FIFO Data (60h...7Ch)	26
3.2.4. Total Latency	9	6. Vital Product Data	27
3.3. Buffer Model	9	6.1. Serial EEPROM Lay-Out	27
3.3.1. Receive FIFO	10	6.2. VPD ID String	27
3.3.2. DMA Buffer	10	6.3. VPD Read-Only Resources	27
3.4. Synchronisation	10	6.4. VPD Read-Write Resources	28
3.5. Buffer Management	11	6.5. Reading VPD Data	29
3.5.1. Ping-Pong, DMA-Driven	11	6.6. Writing VPD Data	29
3.5.2. Start Up	12		

Copyright © 2002-2003 by DEKTEC Digital Video B.V.

DEKTEC Digital Video B.V. reserves the right to change products or specifications without notice. Information furnished in this document is believed to be accurate and reliable, but DEKTEC Digital Video assumes no responsibility for any errors that may appear in this material.

1. General Description

The DTA-120 is a PCI adapter card that reads a DVB/ASI Transport Stream for further processing in software. The transport rate of the incoming MPEG-2 data may have any value between 0 and 214 Mbit/s.

Note

- 214 Mbit/s is the maximum bit rate of a DVB/ASI-compliant Transport Stream.

1.1. Typical Application

The DTA-120 is typically deployed as DVB/ASI input card for MPEG-2 applications running on a PCI-based system. The low cost and high performance of generic computer platforms (e.g. Industrial PC¹) can be leveraged to create cost-effective digital-video solutions.

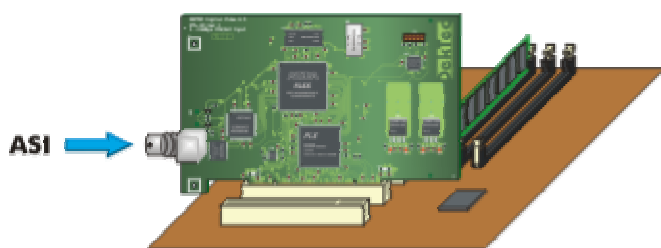


Figure 1. Typical application of the DTA-120 as DVB/ASI input stage in a PC-based digital-video application.

One important class of DTA-120 applications is Transport-Stream analysis:

- MPEG-2 Transport-Stream Analyser,
- MPEG-2 Transport-Stream Monitor,
- PSI/SI Analyser.

Another class of applications is Transport-Stream processing. Using DTA-120 PCI cards, the PC reads one or more MPEG-2 Transport Streams. A software application processes the data and the resulting Transport Stream is output through e.g. the DTA-100 DVB/ASI output card. Examples of such applications include:

- Re-multiplexer,
- Bit-rate transcoder,
- DVB-to-IP gateway,
- Logo inserter,

¹ Application of the DTA-120 is not limited to the PC: Any platform that supports the PCI bus can be used.

- Advertisement inserter with ad-downloading via a Transport Stream.

1.2. Software

The DTA-120 hardware offering includes the following free software:

- WDM device driver for Windows-2000 and Windows-XP,
- Linux device driver,
- DTAPI library, available both for Windows and Linux.

Note

- The Linux device driver and DTAPI are part of the open-source Linux SDK.

The device driver implements “low-level” operations that require direct access to the DTA-120 hardware, such as initiation and coordination of DMA transfers, handling interrupts and reading and writing of Vital Product Data (VPD, §6).

The DTAPI library is a thin layer of user-mode software that packages the driver functions into an easy to use API.

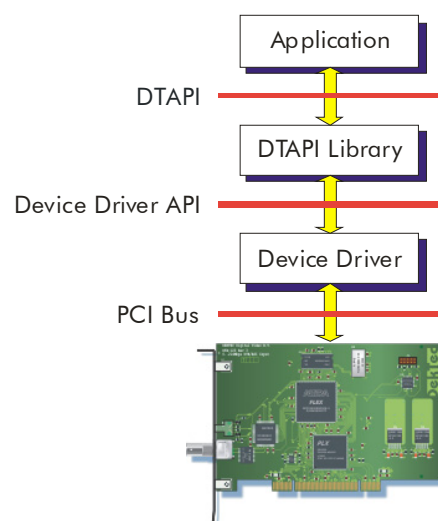


Figure 2. Software stack for the DTA-120.

DEKTEC offers a number of standard applications running on top of (amongst others) the DTA-120:

- DTC-320 *StreamXpert*[™] real-time analyser software;
- *Media Gateway*[™] DVB-to-IP multicast gateway.

Please refer to www.dektec.com for further details about standard software.

DEKTEC's commercial applications use a licensing system that ties the software to a specific DTA-120: the software will only run if an appropriate license code has been programmed into the on-board PROM.

Licenses can be purchased upfront, or the board can be upgraded in the field with the license code.

1.3. Block Diagram

Figure 3 shows a conceptual block diagram of the DTA-120. The DVB/ASI receiver block locks to the serial signal and decodes the DVB/ASI line coding. The result is written to the Receive FIFO, which buffers incoming transport packets until they can be transferred to host memory. The Master-Control block relays the packets to the PCI bus via the PCI-9054 Bus Interface.

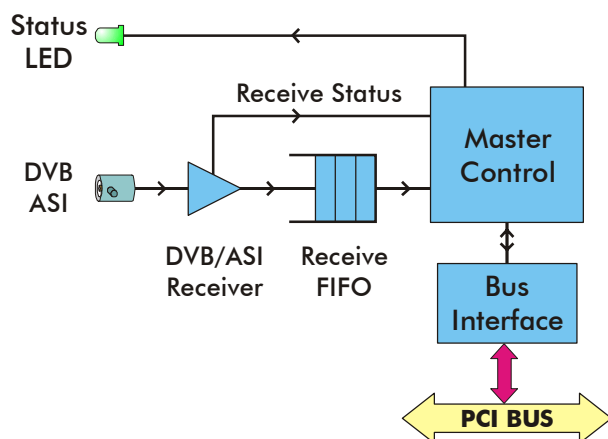


Figure 3. Conceptual block diagram of the DTA-120.

The DVB/ASI receiver provides several status flags and counters that can be accessed programmatically via the DTA-120's register set. A status summary is provided through the Status LED.

1.3.1. Equaliser

The receive circuitry starts with a cable equaliser, which automatically adapts to equalise any cable length from zero to three hundred meters (Belden 8281 or equivalent).

1.3.2. ASI Receive Logic

The equalised signal is processed by a DVB/ASI receiver, which de-serialises the signal, synchronises to the MPEG-2 packet structure and writes the data into the Receive FIFO.

Special features of the ASI Receiver include:

- Time stamping: a sample of a reference clock is attached to each packet, so that the time of entrance of the packet can be derived.
- ASI polarity detection. Inverted DVB/ASI signals are automatically detected and corrected.
- Packet-size conversion, e.g. storage of 188-byte packets, irrespective of the input packet size (188 or 204).

The status of the ASI Receiver is made visible to the outside world through the Receive Status LED.

1.3.3. Receive FIFO

The DTA-120 contains a large (8-MBytes) buffer for incoming Transport-Stream data: the Receive FIFO. A standard SDRAM (as used on computer DIMMs) is used to implement this buffer economically.

The Receive FIFO provides a great deal of freedom for the receiving software. While the application processes data, new transport-stream data is buffered in the Receive FIFO.

1.3.4. PCI-9054 Bus Interface

The DTA-120 uses the PCI-9054 IC made by PLX for interfacing to the PCI Bus. The PCI-9054 also implements the DMA functions required for high-speed streaming of transport packets, with minimal host interaction.

The PCI-9054 supports large host buffers with scatter/gather DMA mode. Such buffers may get fragmented through allocation / de-allocation of memory by OS-components. A scatter/gather list glues the buffer together without requiring software intervention.

Whenever appropriate, this specification provides information on the way the PCI-9054's registers should be used on the DTA-120. To

obtain more details on the operation of the PCI-9054, please refer to the *PCI 9054 Data Book*.

1.3.5. Register Set

Next to the registers in the PCI-9054, the DTA-120 contains a number of dedicated registers in PCI Memory Space (refer to §5 for syntax and semantics). With these Registers, the application software can configure and operate the DVB/ASI-specific monitoring features of the DTA-120.

1.4. References

- *Interfaces for CATV / SMATV Headends and Similar Professional Equipment, DVB DOCUMENT A010 rev.1, May 1997* – This is the original DVB document that specifies physical interfaces for the interconnection of signal processing devices for professional digital-television equipment. One of the interfaces described is DVB/ASI-C. DVB document A010 document has also been issued as *CENELEC EN50083-9*.
- *ISO/IEC 13818-1, Information technology – Generic coding of moving pictures and associated audio information: Systems, April 27th, 1995, also known as “MPEG-2 Systems”* – Specification of the structure of a MPEG-2 Transport Stream.
- *DTAPI: C++ API for DTA-series of Digital-Video PCI-Bus Cards, DEKTEC Digital Video B.V., 2001* – Specification of **DTAPI**: the C++ interface to access the DTA-120 functions at a higher level of abstraction than would be possible using direct device-driver calls.
- *PCI 9054 Data Book, PLX Technology, V2.1, January 2001* – Specification of the PCI 9054, the chip used on the DTA-120 to interface with the PCI bus. Use this document if you need to program the PCI-9054 directly, e.g. when writing a custom device driver. The latest version of this document is available on line at <http://www.plxtech.com>.
- *PCI Local Bus Specification, Revision 2.2, December 18, 1998* – Formal specification

of the protocol, electrical, and mechanical features of the PCI bus.

1.5. Document Overview

This specification describes the details relevant to operating the DTA-120. The information herein is primarily intended for device driver writers and for software developers that have to access the DTA-120 directly from a real-time operating system.

The WDM device driver and DTAPI library encapsulate many programming details of the DTA-120. Users of DTAPI may find this document useful for providing background information, but do not need to master each and every detail.

- *Section 1* introduces the main features of the DTA-120.
- *Section 2* describes the physical interfaces of the DTA-120.
- *Section 3* provides a detailed description of synchronisation and buffer-management, in order to stream data efficiently and reliably.
- *Section 4* lists the PCI Configuration-Space registers supported by the DTA-120.
- *Section 5* describes the *operational registers* on the DTA-120. These registers are used to control and monitor the streaming of digital-video data.
- *Section 6* defines the structure of *Vital Product Data (VPD)* as supported by the DTA-120 and other DEKTEC PCI cards.

2. External Interfaces

2.1. Overview

The lay-out of the DTA-120's PCI bracket is shown in Figure 4 below.

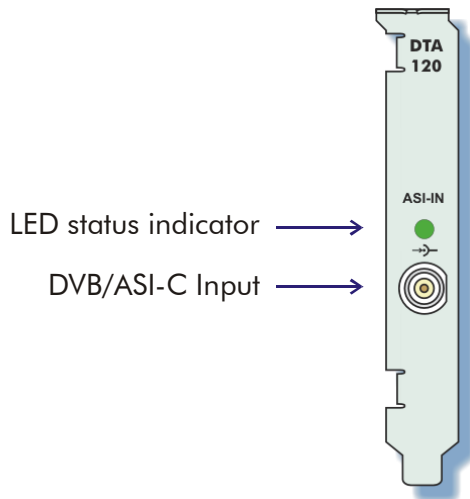


Figure 4. DTA-120 physical interfaces.

The LED indicator shows the status of the DVB/ASI input beneath it.

2.2. DVB/ASI Input

The DVB/ASI input is compliant to the *Asynchronous Serial Interface on coaxial cable (ASI-C)*, as defined in DVB Document A010.

The input connector is 75-Ω BNC.

2.3. LED Status Indicator

The bi-colour (red/green) LED above the input connector displays the status of the DVB/ASI-C input signal.

The status is indicated using a combination of colour-encoding and a flashing pattern. Table 1 shows the various indications.

The flash-pattern pictures show the colour of the LED over time. Grey represents "LED off".

Table 1. LED Status Indications	
LED Status	Meaning
<i>No data</i>	
	No signal
	ASI carrier without TS, or data rate too low (less than about 1 kbps)
<i>Error conditions</i>	
	No lock
	Code violation(s)
	Receive-FIFO overflow
<i>Valid data</i>	
	Valid data but unknown packet size
	Valid 188-byte packets
	Valid 188-byte packets, inverted ASI
	Valid 204-byte packets
	Valid 204-byte packets, inverted ASI
<i>Special</i>	
	Receive-channel reset

Note

- The LED indication may be *overruled* by software.

The *error conditions* "No lock" and "Code violation" may occur in a number of circumstances:

- The input signal is not a DVB/ASI signal, but e.g. a SMPTE-259M signal;
- A badly attached BNC connector;
- The coax cable has a poor quality;
- The cable is too long.

Connecting a cable to the DTA-120 almost certainly leads to a temporary "No lock" (red) indication.

The *valid-data* indications show packet size and inverted-ASI status. If the DVB/ASI input signal is recognised as inverted DVB/ASI, then a very short flash, only just noticeable, interrupts the normal packet-size indication.

2.3.1. LED Power-Up Sequence

Just after power-up, the LED flashes a few times to indicate that the board is initialising. During this short period, the usual meaning of the LED indications does not apply. Instead, the LED pattern shows the board type and the firmware revision. An example start-up pattern is shown in Figure 5 below.

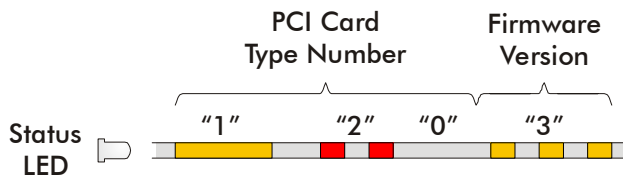


Figure 5. Power-up pattern for DTA-120 with (as example) firmware version 3.

3. Streaming Data

The primary application of the DTA-120 is reading MPEG-2 Transport Packets for processing in a software application. The trickiest part of streaming data is achieving *real-time* operation, or at least – for streams without time stamps – achieving *efficient* operation. A well-balanced buffering scheme is required to compensate for hardware and software latencies.

This section first argues why DMA-based streaming offers superior performance compared to direct reading from the PCI Bus, followed by a description of hardware and software latencies that need to be overcome for real-time streaming. Then the buffer model adopted on the DTA-120 is described. Finally, synchronisation and buffer management are covered.

Using these techniques, §3.5 describes an approach for real-time operation of the DTA-120 in a relatively simple and robust way. This scheme has been implemented in the drivers that come with the DTA-120.

Note

- Of course, other approaches to real-time streaming with the DTA-120 may be feasible as well.

3.1. DMA vs. Direct Reads

The DVB/ASI Transport Stream that enters the DTA-120 is first buffered in the Receive FIFO. Then, the transport packets need to be transferred to a buffer in host memory. The host can do this in one of two ways:

1. The host² directly reads data from the DTA-120's Receive FIFO.
2. The DMA controller on-board of the DTA-120 transfers the data from the Receive FIFO to a DMA Buffer in host memory.

In the first method, the host processor executes a long series of read instructions from the PCI Bus. Effectively, this slows down the processor speed to the PCI-Bus rate. Theoretically the

² This is: the application program or driver running on the host.

read-from-PCI instructions could be interleaved with other instructions, but in practice, this is awkward.

The second method allows the processor to read from main memory, which is at least an order of a magnitude faster than reading from the PCI Bus. The DMA cycles from PCI Bus to main memory– used to transfer the data from the DTA-120 – are invisible to the host processor.

To sum up, the “direct-read” method is simple, but ties the processor to the PCI timing. The DMA method is more complex, yet much faster as a consequence of the vast difference in speed between reading from main memory and reading from the PCI Bus.

The remainder of this section elaborates techniques to transfer data using DMA.

3.2. Latencies

MPEG-2 streams are very sensitive to the loss of even a single packet. Avoiding Receive-FIFO overflow, which leads to packet loss, is essential for flawless operation in most MPEG-2 applications.

This section describes the hardware- and software- latencies that need to be taken into account when streaming MPEG-2 packets with the DTA-120.

3.2.1. PCI-Bus Latency

The DTA-120 shares the PCI Bus with other bus masters that also compete for PCI cycles. The DTA-120 DMA Controller may have to wait a certain amount of time – the *PCI-Bus latency* – before it can acquire the bus and begin a DMA transfer.

Under normal conditions, the maximum duration of PCI latencies is in the order of a few microseconds. On a heavily loaded PCI Bus, latencies can be longer, but practical experience indicates that in all but pathological cases (see note) 2 ms can be safely taken as the absolute maximum PCI-Bus latency.

Note

- Cases are known in which PCI latencies be-

come unbounded³. If real-time streaming is required, it is essential to check the host system for such adversary conditions.

3.2.2. Interrupt Latency

Interrupt latency is the time between a hardware device raising an interrupt and software actually servicing the interrupt.

The DTA-120 hardware/software synchronisation methods (as described below in §3.4) rely on interrupts to signal certain hardware conditions to the software. The maximum interrupt latency has to be taken into account.

3.2.3. Scheduling Latency

The host CPU cannot dedicate all of its time to processing packets coming from the DTA-120: Other threads need processor cycles as well. *Scheduling latency* is the maximum time – in the worst-case scenario – the CPU still has to spend on other jobs, before it can service the DTA-120.

Scheduling latency is hard to grasp. It depends on many factors, like operating system, other software running on the host, relative priorities of threads and other hardware to be serviced.

3.2.4. Total Latency

Adding all up, the total latency is the sum of PCI-Bus-, interrupt- and scheduling- latencies. The total latency is the maximum time elapsing between an event that indicates that data should be transferred, and the time that data is actually copied to host memory.

The principal technique to avoid Receive-FIFO Overflow is to ensure that the total latency is less than the time to completely fill the Receive FIFO.

Thanks to the large size of the Receive FIFO (8 MB), the maximum total latency that can be tolerated is relatively long. At the maximum input rate (214 Mbps), the maximum latency is still 314 ms.

3.3. Buffer Model

Incoming Transport-Stream data is buffered in a cascade of two buffers:

1. The *Receive FIFO*, located on the DTA-120. The DMA Controller on the DTA-120 transfers data bytes from the Receive FIFO to the DMA Buffer.
2. The *DMA Buffer*, located in host memory. The host processor reads data directly from this buffer.

The DMA Buffer should be divided in multiple (sub-)buffers, to avoid contention between host processor and DMA controller⁴. The DMA Controller writes to one DMA Buffer, while the host processes packets from another DMA Buffer.

A scheme with two DMA Buffers is elaborated in §3.5 below. Of course, advanced buffering schemes with more than two DMA Buffers may also be used⁵.

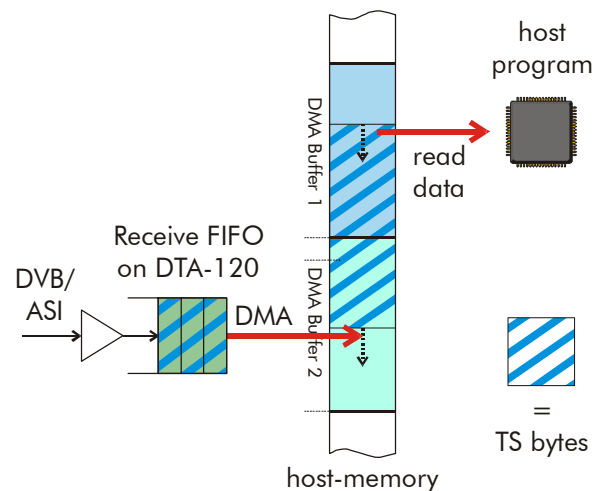


Figure 6. Double-buffer model. The Receive FIFO is implemented in hardware on the DTA-120. The two DMA Buffers are located in host memory.

Figure 6 shows a snapshot of the double-buffer model in action. The buffer is split in DMA Buffer 1 and DMA Buffer 2, together forming one circular buffer. The blue-hatched area

³ E.g. the host CPU writes continuously to a frame buffer on the PCI-Bus for a long period of time.

⁴ It is possible to use a single DMA Buffer, but then CPU read operations from the buffer and DMA write operations to the buffer should be mutually exclusive in time.

⁵ This specification does not provide further details on configurations with more than two DMA Buffers.

represents Transport-Stream data that still has to be processed by the host program.

MPEG-2 data in DMA Buffer 1 is processed by the program running on the host. At the same time, the DMA Controller transfers new packets from the Receive FIFO on the DTA-120 to DMA Buffer 2.

When DMA Buffer 1 has been read empty, and DMA Buffer 2 has been filled completely, the function of both DMA Buffers is swapped.

3.3.1. Receive FIFO

Transport packets entering the DTA-120 are first buffered in the *Receive FIFO*.

The Receive FIFO is implemented on the DTA-120 with an 8-MByte SDRAM. For all practical purposes, the Receive FIFO can be considered a large conventional FIFO that can buffer 8-Mbytes of packet data.

3.3.2. DMA Buffer

A DMA Buffer is an array of bytes allocated in the application's address space. The start- and end- address of a DMA Buffer must be aligned on 4-byte boundaries. The first byte that enters the DTA-120 is stored at relative address 0, the second byte at address 1, etc.

Obviously, the DMA Buffer may not be virtual memory that is swapped out to disk. Either non-paged memory should be used, or the driver should ensure that the pages are locked into physical memory whenever the DTA-120's DMA Controller may write to them.

A DMA Buffer maps to a contiguous address range in virtual-address⁶ space. The DMA Buffer needs not be contiguous in physical-address space: Memory pages may be *scattered* over physical memory⁷. The PCI-9054 Scatter/Gather DMA mode can be used to transfer such a scattered DMA Buffer in one go, without requiring processor intervention to glue pages together.

⁶ It is not a strict requirement that the DMA Buffer is contiguous in virtual address space. Nonetheless, application programmers will find it very convenient.

⁷ This means that the DMA Buffer may be allocated from a fragmented memory pool.

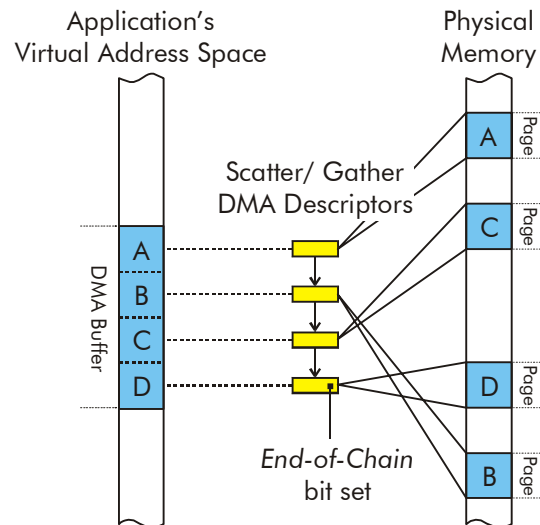


Figure 7. The DMA Buffer appears contiguous to the application, while pages are scattered over physical memory. Scatter/gather DMA allows writing to many pages in the DMA Buffer in one go without processor intervention.

Scatter/Gather DMA uses a list of *Scatter/Gather DMA Descriptors* stored in (non-paged) host memory. This so-called *scatter/gather list* can be built at the same time as the DMA Buffer is allocated.

The last descriptor in the list shall have its *End-of-Chain* bit set. The corresponding interrupt can be enabled, so that the driver is alerted when the DMA Buffer has been filled.

The descriptor syntax and the way to initialise and operate scatter/gather DMA are described in the PCI-9054 data book. Note that the DTA-120 uses demand-mode DMA. This implies that DMA channel 0 shall be used for performing the DMA transfers.

Note

- The scatter/gather mechanism incurs a little overhead per descriptor. Therefore, scatter/gather buffers should not be made too small, as this will lead to degraded performance. Buffers with the size of a memory page are fully acceptable.

3.4. Synchronisation

The DTA-120 reads Transport Packets, while the host processor processes them. Obviously,

packet reception must be synchronised to packet processing, or discontinuities will occur.

The DTA-120 hardware is tailored for *DMA-Driven* synchronisation: The host software locks packet processing to the completion of DMA transfers. Refer to §3.5.1 for how this works in combination with buffer management.

DMA transfers from Receive FIFO to DMA buffer can and should be initiated before the Receive FIFO contains sufficient data to complete the DMA transfer. In this manner, the DMA-done interrupt can be used to synchronise the host software to the incoming DVB/ASI Transport Stream: The DMA-done interrupt fairly accurately represents the moment in time that another buffer load with Transport Packets has been read by the DTA-120.

DMA-driven synchronisation works reliably because the DTA-120 hardware implements *demand-mode* DMA: DMA transfers are requested on the PCI Bus only as long as the DTA-120's Receive FIFO has data available. When the FIFO becomes empty, the DMA process stalls. When new data enters the Receive FIFO again, DMA resumes.

In other words: The Receive FIFO cannot underflow in DMA-driven operation. The handshaking hardware prevents this from happening.

3.5. Buffer Management

This section discusses how to manage DMA Buffers such that synchronisation of packet reception by the DTA-120 and packet processing by the host is achieved.

3.5.1. Ping-Pong, DMA-Driven

As explained in §3.3, efficient streaming of data to the DTA-120 requires at least two DMA Buffers. The DMA Controller on the DTA-120 writes packets from the Receive FIFO to one buffer. At the same time, the host program reads and processes packets from the other buffer. When both DMA is done and the packets in the other buffer have been processed completely, the DMA Buffers swap function. This process continues ad infinitum.

Note

- An advanced buffer-management scheme may use more than two DMA Buffers. However, for the majority of applications a double-buffering will suffice.

The use of two buffers that swap function after each cycle – also known as *Ping-Pong* buffering – is illustrated in Figure 8.

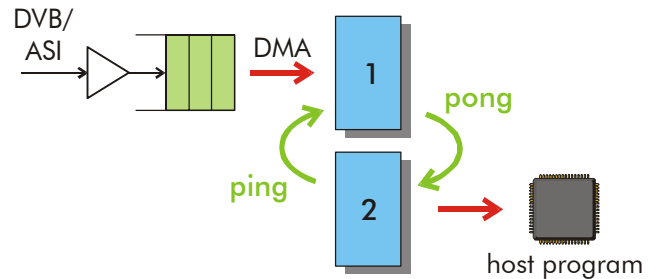


Figure 8. Ping-Pong buffering. The DMA Controller writes packets from the Receive FIFO into one buffer, while the host reads packets from the other buffer. When both are finished, the "Ping-Pong" swap is executed.

DMA-Driven flow control in a double-buffering scheme is illustrated in the message-sequence chart shown in Figure 9.

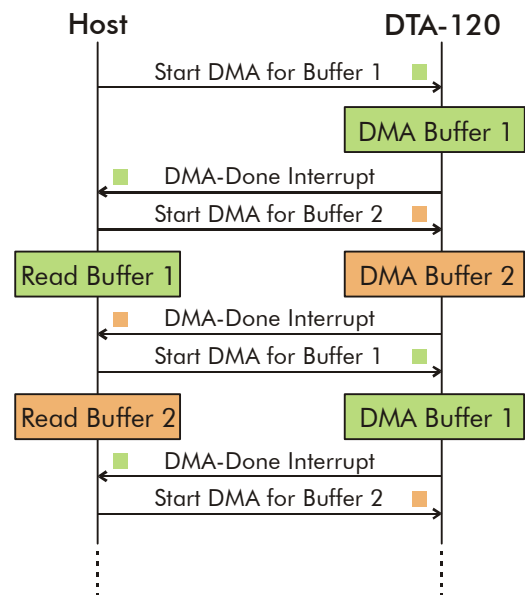


Figure 9. Ping-Pong buffer management using DMA-Driven flow control. After the host has read a buffer and the DMA write to the other buffer is done, the buffers swap function.

The DMA-Done Interrupt is the handshake signal. It triggers the host to read and process new packets, and to initiate a new DMA transfer.

Note

- While waiting for the DMA-Done Interrupt, it is opportune for a device driver to sleep the process and give another process a chance to run.

It is instructive to ponder on the limiting factor in the Ping-Pong process: Host or DMA?

DMA has to be the limiting factor. The effective average rate of DMA is equal to the receive rate. The host has to be able to read and process packets faster than packets enter the DTA-120, otherwise long-term operation cannot be sustained. So, on average the host finishes reading a buffer before DMA to the other buffer is done.

3.5.2. Start Up

The following procedure is recommended to start-up the Ping-Pong process:

1. Allocate DMA Buffers, allocate and initialise Scatter/Gather DMA Descriptors.
2. Reset the DTA-120. This will clear the Receive FIFO and reset the Receive-Control field (§5.2.1.1) to **Idle**.
3. Initiate the first DMA transfer from Receive FIFO to DMA Buffer.
As the Receive FIFO is still in idle mode, no data will actually be transferred yet.
4. Set the Receive-Control field to **Rcv**.
At this time, Transport Packets are allowed to enter the Receive FIFO. Shortly after, the first data will be transferred to the DMA Buffer.
5. Normal Ping-Pong Operation has been entered.

3.6. Packet Alignment

The DTA-140 supports alignment of transport packets in the DMA buffer:

- MPEG-2 sync bytes (0x47) are stored at 32-bit aligned byte positions;
- After setting Receive-Control to **Rcv**, the first

data transferred to the DMA Buffer will start at a packet boundary.

Packet alignment is provided for in Receive Modes **St188**, **St204** and **StMp2**, but not in **StRaw**. In raw mode, the DTA-140 does not take the packet structure into account, and sync bytes may appear at any relative address.

3.6.1. Fixed Buffer Structure

Packet alignment opens up the possibility of a fixed packet lay-out in the DMA Buffer(s). Hereto, the Receive Mode must be set to **St188** or **St204**, and the buffer size must be chosen a multiple of the packet size.

Figure 10 shows an example of packet alignment in an 1880-byte DMA-Buffer (10 packets), with Receive Mode set to **St188**.

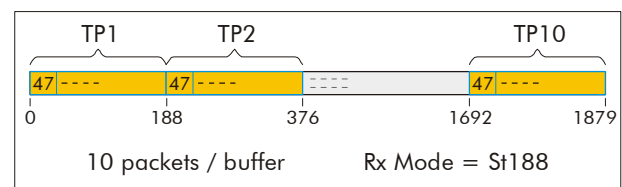


Figure 10. Transport packets stored at fixed locations. The buffer size must be a multiple of the packet size.

Notes

- Receive Mode **StMp2** cannot be used reliably for packet alignment in a DMA Buffer: When the size of the incoming transport packets changes dynamically, alignment will likely break.
- It is good software-engineering practice to test for packet alignment, even if packets "should always be aligned". If the test would fail, the input channel should be reset and input processing restarted.

3.6.2. Switching Receive Mode

The DTA-140 supports *dynamic* switching of Receive Mode, this is going from one mode to another while, at the same time, data is being stored in the Receive FIFO. However, such dynamic switching may break packet alignment.

Note

- Dynamic switching may even break 32-bit alignment of MPEG-2 sync bytes.

If you want to switch Receive Mode, while reliably maintaining packet alignment (and 32-bit alignment), a channel reset is required:

1. Set Receive Control to **Idle**;
2. Reset receive channel;
2. Set Receive Mode to the desired value;
3. Set Receive Control to **Rcv**.

An inevitable side effect of the use of a channel reset is the "loss" of a number of incoming packets.

3.7. Time Stamping

This section discusses the use of time stamping, as a method to measure the time of entrance of incoming transport packets. The measurement is made available to the application in the form of *time stamps*, attached to the packets.

Applications of time-stamping include:

- Real-time processing of transport streams, e.g. for PCR correction.
- Synchronisation of multiple incoming transport streams in (re-)multiplexers.
- Measurement of real-time packet-delivery jitter.

3.7.1. Block Diagram

Time stamps are derived from a *reference clock counter*, running at 40.5 Mhz. Every time a packet enters the DTA-140, a sample of the counter is taken and the value is prepended to the packet.

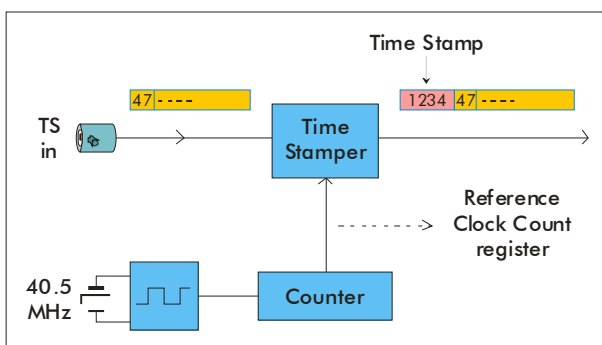


Figure 11. Schematic block diagram of the time-stamping hardware.

The value of the reference-clock counter can also be observed from the PCI bus, by reading the Reference-Clock-Count register (§5.1.4).

3.7.2. Time-Stamp Format

Time stamping can be enabled by setting the `RxTimeStamp` the (§5.2.1.4) in the Receive-Control register. Receive Mode must be one of **St188**, **St204** or **StMp2**.

Note

- Time stamping is not supported in raw mode (**StRaw**).

Time stamps are stored in Little-Endian format into the 4 bytes that are placed before the packet, as shown in Table 2.

Addr ⁸	Value	Description
00h	bit 7..0	Time stamp in 4 bytes
01h	bit 15..8	
02h	bit 23..16	
03h	bit 31..24	
04h	47h	MPEG-2 Sync
:	:	Transport-packet header and payload
BFh	XX	Last byte of packet

The application program can find the location of the time stamps by synchronising to the MPEG-2 Sync bytes.

Another option is to choose a buffer size that is a multiple of the packet + time-stamp size (e.g. a multiple of 192 for **St188**). The first four bytes of the buffer will then always contain a time stamp. See also §3.6.1.

⁸ Relative address in packet.

4. Configuration Space

The DTA-120 acts as a single logical PCI Bus device. It implements the configuration registers required for identifying the device, control PCI Bus functions, and provide PCI Bus status.

Table 4 displays the address map of registers defined in configuration space:

- Black fields indicate configuration registers

supported by the DTA-120.

- Red-text cells represent registers supported by the PCI bridge chip, but not used for operating the DTA-120.
- Grey-text cells represent registers defined in the PCI Local Bus Revision 2.2 specification, but not supported on the DTA-120.

Table 3. Configuration Space – Address Map

Address Offset	Byte			
	3	2	1	0
00h	Device ID		Vendor ID	
04h	Status Register		Command Register	
08h	Class Code			Revision ID
0Ch	BIST	Header Type	Latency Timer	Cache Line Size
10h	PCI Base Address 0; used for memory-mapped configuration registers (PCI 9054)			
14h	PCI Base Address 1; not used			
18h	PCI Base Address 2; used for memory-mapped operational registers ⁹			
1Ch	PCI Base Address 3; not used			
20h	PCI Base Address 4; not used			
24h	PCI Base Address 5; not used			
28h	Card Bus CIS Pointer; not supported			
2Ch	Subsystem ID		Subsystem Vendor ID	
30h	Expansion ROM Base Address Register; not used			
34h	Reserved			Next_Cap = 40h
38h	Reserved			
3Ch	Maximum Latency	Minimum Grant	Interrupt Pin	Interrupt Line
40h	Power Management Capabilities; not used		Next_Cap = 48h	Capability ID = 01h
44h	Power Management Register; not used			
48h	Hot Swap; not used		Next_Cap = 4Ch	Capability ID = 06h
4Ch	VPD; used for manufacturing / service		Next_Cap = 00h	Capability ID = 03h
50h	VPD; used for manufacturing / service			

Table 4 shows a short description of the registers in configuration space.

⁹ Operational registers are mapped in "Local Address Space 0" of the PCI 9054.

Table 4. Configuration Space – Register Overview

Register	Bits	RW*	Value	Short Description
Vendor ID	16	R	10B5h	Identifies PLX as manufacturer of the PCI interface chip.
Device ID	16	R	9054h	Identifies the PCI interface chip (PCI 9054).
Command Register	16	RW	-	Provides coarse control on the ability to generate and respond to PCI cycles.
Status Register	16	RWC	-	Status of PCI-Bus relevant events.
Revision ID	8	R	0	Revision number of your DTA-120.
Class Code	24	R	FF0000h	Generic function of the DTA-120.
Cache Line Size	8	R	16	System cache line size in units of 32-bit words.
Latency Timer	8	RW	-	Amount of time in PCI-Bus-clock units that the DTA-120 may retain ownership of the PCI Bus.
Header Type	8	R	0	Specifies layout of configuration addresses 10h through 3Fh and single / multiple functions.
BIST	8	R	0	PCI Built-In Self Test (BIST).
PCI Base Address 0	32	RW	-	Memory attributes and base memory address for memory accesses to PCI-9054 registers
PCI Base Address 2	32	RW	-	Memory attributes and base memory address for memory accesses to <i>Local Address Space 0</i> , which is used to access the DTA-120's operational registers (Refer to §5).
Subsystem Vendor ID	16	R	14B4h	Identifies the manufacturer of the DTA-120. Subsystem Vendor ID and Subsystem Device ID are leased from Philips BE.
Subsystem Device ID	16	R	D114h	Identifies the PCI card as a DTA-120.
Interrupt Line	8	RW	-	Interrupt line routing information.
Interrupt Pin	8	R	01h	Interrupt pin used by the DTA-120.
Minimum Grant	8	R	10h	Length of time (in 250-ns units) the DTA-120 would like to retain master ship of the PCI Bus.
Maximum Latency	8	R	1Ah	Frequency in which the DTA-120 would like to gain access to the PCI Bus.

5. Target Address Space

The DTA-120's operational registers are mapped in Local-Address Space 0 of the PCI 9054. The PCI Base address of these registers is specified in BAR2. All accesses to the operational registers shall be 32-bit transfers.

Table 5. Operational Registers – Memory Map									
Address Offset	Byte								
	3		2		1		0		
General									
00h	General Control								
04h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	General Status		
08h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0	Programming	
0Ch	Reference-Clock Count								
10h ... 1Fh	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
Receive (Rx)									
20h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
24h	0 0 0 0	0 0 0 0	0 0	Receive Control					
28h	0 0 0 0	0 0 0 0	0 0 0 0	0 0	Receive Status				
2Ch ... 34h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
38h	0 0 0 0	0 0 0 0	Receive FIFO Load						
3Ch	Receive Diagnostics								
40h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	Rx Loop-Back Data		
44h ... 4Ch	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
50h	Receive Valid Count								
54h	Receive Violation Count								
58h ... 5Ch	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
60h ... 7Ch	Receive FIFO Data								

Note

- The register lay-out shown in Table 5 has been introduced in DTA-120 firmware version 4 and is incompatible to prior firmware versions. DEKTEC recommends upgrading DTA-120s with firmware prior to version 4, using the *DtInfo* utility.
- The General- and Receive- register blocks in the DTA-120 are compatible with the corresponding register blocks on the DTA-140.

Table 6. Operational Registers – General Registers

Register	Field	Bit Pos	#	RWC*	Short Description
General Control	PE	0	1	RW	Serial EEPROM Program Enable
	PRE	1	1	RW	Serial EEPROM Protect Register Enable
	Reset	2	1	W	Reset DTA-120 circuitry
	<i>reserved</i>	3	1	R	Not used
	PerIntEn	4	1	RW	Periodic-Interrupt Enable
	<i>reserved</i>	7...5	3	R	Not used
	FirmwRev	15...8	8	R	Firmware Revision
	TypeNum	23...16	8	R	Type Number: 120 for DTA-120
	LedControl	24	1	RW	Take over LED Control
	LedGreen	25	1	RW	State of Green LED
	LedRed	26	1	RW	State of Red LED
General Status	<i>reserved</i>	3...0	4	R	Not used
	PerInt	4	1	RC	Periodic Interrupt
Programming	TRST	0	1	RW	Control of TRST# pin
	TDO	1	1	R	Status of EPC2 TDO pin
	TMS	2	1	W	Control of EPC2 TMS pin
	TCK	3	1	RW	Control of EPC2 TCK pin
	TDI	4	1	RW	Control of EPC2 TDI pin
	ProgramEpc	5	1	RW	Program EPC2
Reference Clock	RefClkCnt	31...0	32	R	Reference-Clock Count

Table 7. Operational Registers – Receive Registers

Register	Field	Bit Pos	#	RWC*	Short Description
Receive Control	RxMode	1...0	2	RW	Receive Mode: St188/St204/...
	RxAsiInv	3...2	2	RW	Invert ASI input: Auto/Normal/Invert
	<i>reserved</i>	4	1	R	Not used
	RxCtrl	5	1	RW	Receive Control: Idle/Rcv
	<i>reserved</i>	6	2	R	Reserved for expansion of RxCtrl
	RxTimeStamp	7	1	RW	Insert Time Stamps before packets
	<i>reserved</i>	8	1	R	Not used
	RxOvfIntEn	9	1	RW	Receive-FIFO Overflow Interrupt Enable
	RxSyncIntEn	10	1	RW	Synchronisation-error Interrupt Enable
<i>reserved</i>	18...11	8	R	Not used	

Table 7. Operational Registers – Receive Registers

Register	Field	Bit Pos	#	RWC*	Short Description
Receive Control (cnt'd)	RxLedControl	19	1	RW	Take over Rx-LED Control
	RxLedGreen	20	1	RW	State of Green Rx LED
	RxLedRed	21	1	RW	State of Red Rx LED
	RxClrFifo	22	1	RW	Clear Receive FIFO
	RxRst	23	1	RW	Reset Receive Channel
Receive Status	RxPckSize	1...0	2	R	Packet size: Rx188/Rx204/RxInv
	<i>reserved</i>	3...2	2	R	Not used
	RxSdramSize	7...4	4	R	SDRAM Size = Size of Receive FIFO
	<i>reserved</i>	8	1	R	Not used
	RxOvfInt	9	1	RC	Receive-FIFO Overflow Interrupt
	SyncInt	10	1	RC	Synchronisation-error Interrupt
	<i>reserved</i>	13...11	3	R	Not used
	RxAsiCD	14	1	R	DVB/ASI Carrier Detect
	RxAsiLock	15	1	R	Locked to DVB/ASI input signal
	RxRateOk	16	1	R	Input Rate "Ok" (not too low)
	RxAsiInv	17	1	R	Invert DVB/ASI input signal – status
Receive FIFO Load	RxFifoLoad	23...0	24**	R	Current Load of Receive FIFO in #bytes
Receive Diagnostics	<i>reserved</i>	7...0	8	R	Not used
	RxLoopBack	8	1	RW	Loop-back mode
	<i>reserved</i>	9	1	R	Not used
	RxRfRateOvf	10	1	R	Receive-FIFO Rate Overflow
	RxSdFull	11	1	R	SDRAM Full
Rx Loop-Back Data	RxLbData	7..0	8	W	Write data to Receive FIFO
Rx Valid Count	RxValidCnt	31...0	32	R	Sample of DVB/ASI Valid-byte Counter
Rx Violation Count	RxViolCnt	31...0	32	R	Sample of Violation Counter
Receive-FIFO Data	RxFifoData	31...0	4x8	R	Transport-Stream data: 4 bytes at a time

* R=Readable, W=Writeable, C=Clearable (clear when a '1' is written to bit position).

** Number of bits depends on RxSdramSize. Shown size (24-bits) is valid for 8-Mbyte SDRAM.

5.1. General Registers

The General Registers contain control and status bits that are not directly related to the DVB/ASI input- and output- channel on the DTA-120.

The lay-out of the General Registers is similar to that of other DTA-1xx PCI adapter cards, to simplify the writing of generic device drivers that can handle multiple DEKTEC cards.

5.1.1. General Control (00h)

The lay-out of the General-Control register is shown in Table 8.

Bit	Mnem	Description
0	PE	Serial EEPROM Program Enable
1	PRE	Serial EEPROM Protect Register Enable
2	Reset	Soft Reset
3	<i>reserved</i>	
4	PerIntEn	Periodic-Interrupt Enable
7...5	<i>reserved</i>	
15...8	FirmwRev	Firmware Revision
23...16	TypeNum	Value=120
24	LedControl	Take over LED Control
25	LedGreen	State of Green LED
26	LedRed	State of Red LED
Bits 31...27 of this register are tied to '0'.		

5.1.1.1. PE – Program Enable

The Program-Enable field directly controls the PE signal of the serial EEPROM. The EEPROM can only be programmed when this bit is set to '1'. During normal operation, PE should remain '0'.

5.1.1.2. PRE – Protect Register Enable

The Protect-Register-Enable field directly controls the PRE signal of the serial EEPROM. It enables the write-protection mechanism in the EEPROM. During normal operation, this field should remain '0'.

Warning

- Issuing a write-protection command to the serial EEPROM is an irreversible operation. Incautious use of the PRE bit may destroy the Vital-Product Data read/write capability!

5.1.1.3. Reset – Software Reset

Writing a '1' to the Reset bit issues a "soft" reset to the DTA-120. The following fields and logic circuitry are affected:

- RxCtrl in the Receive-Control register is reset to **TxIdle**.
- The entire contents of the Receive FIFO are cleared.
- The Receive-FIFO-Load register is reset to zero.
- The Interrupt-Status flags in the Receive-Status register are cleared.
- A number of internal state machines are reset.

Other fields in the operational registers are not affected, notably:

- Receive Mode (RxMode) in the Receive-Control register.
- Loop-Back Mode in the Diagnostics register.
- Interrupt-Enable bits in the Receive-Control register: interrupts that were enabled remain enabled.
- PE and PRE in the General-Control register.

This behaviour is by design, so that the data pipelines in the DTA-120 can be reset without compromising other processes running on the PCI card.

The Reset bit is write-only. The write operation triggers the reset action: it is not required to reset the bit to '0' again. The next time a '1' is written to the Reset bit, the board will be reset again.

5.1.1.4. PerIntEn – Periodic Interrupt Enable

Writing a '1' to this bit enables¹⁰ the *Periodic Interrupt* (§5.1.2.1).

¹⁰ To actually enable the Periodic Interrupt on the PCI Bus, the Local-Interrupt-Input-Enable bit in the PCI9054's Interrupt Control/Status register must also be set to '1'.

5.1.1.5. FirmwRev – Firmware Revision

This read-only field identifies the current revision level of the DTA-120's firmware.

Note

- The Firmware Revision level is independent of the DTA-120 board revision (which can be read from VPD).

5.1.1.6. TypeNum – Type Number

The Type-Number field identifies the board in a straightforward way. For the DTA-120, the field's value is fixed to **120** (decimal).

Note

- Apart from this field, the board's type number is also encoded in the Vital Product Data (VPD), which is the primary source of descriptive data about a board. The purpose of the Type-Number field is to provide a convenient way for device drivers to distinguish between different kinds of DTA-1xx boards at start-up.

5.1.1.7. LedControl – Take over LED Control

When this field is set to '0', the state of the bi-colour LED indicator on the PCI bracket is determined by the hardware, as described in §2.3.

When this field is set to '1', the receive hardware is "disconnected" from the LED indicator and the LED is controlled directly by fields LedGreen and LedRed.

This bit field is reset to '0' upon a hardware- or software reset.

5.1.1.8. LedGreen – State of Green LED

When LedControl is '1', this field controls the **green** colour of the bi-colour LED next to the connector on the PCI bracket.

5.1.1.9. LedRed – State of Red LED

When LedControl is '1', this field controls the **red** colour of the bi-colour LED next to the connector on the PCI bracket.

5.1.2. General Status (04h)

The lay-out of the General-Status register is shown in Table 9.

Table 9. General-Status Register – Format

Bit	Mnem	Description
3...0	<i>reserved</i>	
4	PerInt	Periodic Interrupt
Bits 31...5 of this register are tied to '0'.		

5.1.2.1. PerInt – Periodic Interrupt

When set to '1', this bit indicates that the *Periodic Interrupt* is pending. The Periodic Interrupt is generated automatically every 2^{21} clock cycles of the on-board 40.5-MHz reference clock. This corresponds to approximately once every 51.8 ms, or 19.31 times per second.

5.1.3. Programming (08h)

The Programming Register can be used to update the firmware of the DTA-120 ("flashing"). The fields are connected to the JTAG programming lines of the EPC2 EEPROM

Table 10. Programming Register – Format

Bit	Mnem	Description
0	TRST	Control of TRST# pin
1	TDO	Status of TDO pin
2	TMS	Control of TMS pin
3	TCK	Control of TCK pin
4	TDI	Control of TDI pin
5	PerInt	Periodic Interrupt
Bits 31...6 of this register are tied to '0'.		

5.1.4. Ref. Clock Count (0Ch)

The Reference-Clock-Count register provides access to the DTA-120's *reference-clock*, which is a free-running counter at a rate of 40.5 MHz.

Table 11. General-Status Register – Format

Bit	Mnem	Description
31...0	RefClkCnt	Reference Clock Count

The hardware uses the reference clock for the generation of time stamps for incoming transport packets.

The Reference-Clock-Count register enables tracking of the 40.5-MHz clock in software, e.g. to correlate the input time stamps of multiple receive boards.

5.2. Receive Registers

5.2.1. Receive Control (24h)

The Receive-Control register contains a number of fields that allow the device driver to control receive-specific functions of the DTA-120.

Table 12. Receive-Control Register – Format

Bit	Mnem	Description
1...0	RxMode	Receive Mode
3...2	RxAsiInv	Invert ASI-input control
4	<i>reserved</i>	
5	RxCtrl	Receive Control
6	<i>reserved</i>	
7	RxTimeStamp	Insert Time Stamps
8	<i>reserved</i>	
9	RxOvfIntEn	Receive-FIFO-Overflow-Interrupt Enable
10	RxSyncIntEn	Synchronisation-error-Interrupt Enable
18...11	<i>reserved</i>	
19	RxLedControl	Take over LED Control
20	RxLedGreen	State of Green LED
21	RxLedRed	State of Red LED
22	RxClrFifo	Clear Receive FIFO
23	RxRst	Reset Receive Channel
Bits 31...24 of this register are tied to '0'.		

5.2.1.1. RxMode – Receive Mode

Receive Mode is a 2-bit field that controls the processing applied to incoming packets.

Value	Mode	Definition
00	St188	Store 188-byte packets.
01	St204	Store 204-byte packets.
10	StMp2	Store 188- or 204-byte packets.
11	StRaw	No notion of packets. All incoming data is stored in the Receive FIFO.

The default Receive Mode is **St188**. In this mode the DTA-120 accepts 188- and 204-byte packets, but always stores 188 bytes per packet. If the input contains 204-byte packets, the 16 trailing bytes are dropped, irrespective of their content. Incoming data without MPEG-2 packet structure is dropped entirely.

Receive Mode **St204** is similar to **St188**, but now 204 bytes are stored per packet. If the input contains 188-byte packets, 16 zero bytes are appended.

In Receive Mode **StMp2**, MPEG-2 packets are stored as they enter the system. However, if the DTA-120 cannot synchronise to a packet structure in the incoming data, input data is dropped.

Finally, in Receive Mode **StRaw** all data bytes are stored in the Receive FIFO, without MPEG-2 synchronisation.

Notes

- Receive Mode can be switched to another mode *dynamically*; this is while data is being stored in the Receive FIFO. A few packets may get lost in the resynchronisation process.
- Dynamically switching Receive Mode in modes **St188**, **St204** and **StMp2** may break 32-bit alignment and packet alignment. Refer to §3.6.2 for additional information.

5.2.1.2. RxAsiInv – Invert DVB/ASI-Input Control

Invert-DVB/ASI-Input Control is a 2-bit field that controls whether the DTA-120 attempts to automatically detect the polarity of the DVB/ASI

input signal, or forces non-inverted or inverted usage of the input signal.

Value	Mnem	Definition
00	Auto	Auto-detect polarity of DVB/ASI input signal.
10	Normal	Do not invert DVB/ASI input
11	Invert	Invert DVB/ASI input signal

The DVB/ASI signal is sensitive to signal polarity. Without corrective measures, an inverted DVB/ASI signal (which may be caused by an inverting distribution amplifier) will be decoded incorrectly by a standard DVB/ASI receiver.

The DTA-120 contains circuitry to automatically detect whether a signal is inverted. Hereto, the card first tries to lock on 188/204-byte packets with a non-inverted signal. If synchronisation cannot be achieved within about 10 packets, the DTA-120 tries again with the input signal inverted. This cycle continues until synchronisation is achieved.

In certain circumstances, e.g. if it is known that the incoming signal does not contain 188-byte or 204-byte MPEG-2 packets, automatic detection of signal polarity is undesirable. In such cases, Manual-Invert Control should be set to **Normal** or **Invert**, so that the DTA-120 will not arbitrarily switch between inverted and non-inverted modes.

5.2.1.3. RxCtrl – Receive Control

The Receive-Control field controls storage of data into the Receive FIFO.

Value	Mnem	Definition
0	Idle	No new data is stored in Receive FIFO.
1	Rcv	Store incoming data into Receive FIFO.

After a power-up condition, Receive Control is initialised to **Idle**. Whenever Receive Control is set to **Idle**, the input circuitry is “disconnected” from the Receive FIFO and no new data can be stored in the Receive FIFO.

When Receive Control is set to **Rcv**, actual storage of transport packets in the Receive FIFO begins.

5.2.1.4. RxTimeStamp – Insert Time Stamps

The Time-Stamp field controls whether incoming packet are tagged with a 32-bit time stamp.

Value	Mnem	Definition
0	NoTs	Do not insert time stamps.
1	Stamp	Insert time stamps.

Time stamps are derived from the 40.5-MHz reference clock counter. The value of this counter is stamped at the moment that the MPEG-2 sync byte enters the DTA-120 (with a delay of less than 1 μ s).

Please refer to §3.7.2 for a description of the format of time stamp.

Insertion of time stamps is not available in Receive Mode **StRaw**.

5.2.1.5. RxOvfIntEn – Receive-FIFO Overflow Interrupt Enable

Writing a ‘1’ to this bit enables the Receive-FIFO-Overflow Interrupt (§5.2.2.3).

5.2.1.6. RxSyncIntEn – Synchronisation-Error Interrupt Enable

Writing a ‘1’ to this bit enables the Synchronisation-Error Interrupt (§5.2.2.4).

5.2.1.7. RxLedControl – Take over LED Control

When this field is ‘0’, the state of the bi-colour LED indicator on the PCI bracket is determined by the hardware, as described in §2.2.

When this field is ‘1’, the hardware is disconnected from the LED indicator. Instead, the LED is controlled directly by fields `LedGreen` and `LedRed`.

This bit is reset to ‘0’ upon a hardware- or software reset.

5.2.1.8. RxLedGreen – State of Green LED

If `LedControl` is '1', this field controls the **green** colour of the bi-colour LED next to the connector on the PCI bracket.

5.2.1.9. RxLedRed – State of Red LED

If `LedControl` is '1', this field controls the **red** colour of the bi-colour LED next to the connector on the PCI bracket.

5.2.1.10. RxClrFifo – Clear Receive FIFO

Writing a '1' to this field¹¹ clears the contents of the Receive FIFO and resets a number of related control fields:

- The FIFO-Load register is cleared to zero.
- The Receive-FIFO Overflow Interrupt flag is cleared.
- `RxCtrl` in the Receive-Control register is reset to **Idle** (to avoid that new data is immediate written in the Receive FIFO again).

5.2.1.11. RxRst – Reset Receive Channel

Writing a '1' to this field¹¹ resets the DVB/ASI Receive Channel.

The following fields and logic circuitry are affected by a Reset Transmit Channel action:

- The actions brought about by `RxClrFifo`: clearing the Receive FIFO and the Receive - FIFO Overflow flag, setting Receive Control to **Idle**.
- The Synchronisation-Error Interrupt flag is cleared.
- The state machines used in the Receive-Channel hardware are reset.

Other fields in the operational registers are not affected, notably:

- Receive Mode (`RxMode`).
- Invert DVB/ASI-Input Control (`RxAsiInv`).
- Insert Time Stamps (`RxTimeStamp`).
- Receive Loop-Back Mode in the Diagnostics register.
- Interrupt-Enable bits in the Receive-Control register: interrupts that were enabled remain enabled.

¹¹ The `RxClrFifo` and `RxRst` bits are write-only. The write operation triggers the clear action: it is not required to reset the bit to '0' again.

5.2.2. Receive Status (28h)

The Receive-Status register contains a number of fields that allow the device driver to read status information from the DTA-120.

Table 17. Receive-Status Register – Format		
Bit	Mnem	Description
1...0	<code>PckSize</code>	Size of incoming packets
3...2	<i>reserved</i>	
7...4	<code>RxSdramSize</code>	SDRAM Size
8	<i>reserved</i>	
<i>Interrupt status flags</i>		
9	<code>OvfInt</code>	Receive-FIFO Overflow
10	<code>RxSyncInt</code>	Synchronisation error
13...11	<i>reserved</i>	
<i>Other status flags</i>		
14	<code>RxAsiCD</code>	DVB/ASI Carrier Detect
15	<code>RxAsiLock</code>	Locked to DVB/ASI signal
16	<code>RxRateOk</code>	Input Rate "Ok"
17	<code>RxAsiInv</code>	Invert ASI signal – status
Bits 31...18 of this register are tied to '0'.		

The interrupt-status flags (bits 9 and 10) in this register share common behaviour:

- An interrupt-status flag is set when the corresponding condition occurs. The flag remains set until it is explicitly cleared.
- Writing a '1' to the flag clears the interrupt-status flag, and also clears the PCI interrupt¹² (unless another interrupt condition is pending).
- The interrupt-status flag only leads to an interrupt if the corresponding interrupt-enable bit in the Receive-Control register is set, and interrupts in the PCI 9054 have been enabled.
- The operation of the interrupt-status bits is independent from the state of the interrupt-enable bit: If the interrupt-enable bit is '0', the interrupt-status flag still latches the corresponding condition.

¹² No write action to a PCI-9054 register is required.

5.2.2.1. RxPckSize – Packet Size

Packet Size is a 2-bit field that indicates the length of the packets currently being received on the Transport-Stream input.

Value	Mode	Definition
00	RxInv	Invalid packet size.
01	-	<i>reserved</i>
10	Rx188	Receiving 188-byte packets.
11	Rx204	Receiving 204-byte packets.

Packet-Size values **Rx188** and **Rx204** indicate that the DTA-120 receives correctly formatted packets of 188 or 204 bytes respectively.

Packet-Size value **RxInv** indicates that the DTA-120 is currently out of synchronisation with the input Transport Stream.

5.2.2.2. RxSdramSize – SDRAM Size

SDRAM Size is a static read-only field that indicates the size of the SDRAM on-board of the DTA-120. The SDRAM size determines the maximum size of the Receive FIFO.

Value	Size	Comment
0000	8 MB	Minimum supported size.
0001	16 MB	May be supported in future revisions of the DTA-120.
0010	32 MB	May be supported in future revisions of the DTA-120.
other	<i>reserved</i>	

5.2.2.3. RxOvfInt – Receive-FIFO Overflow Interrupt

When set to '1', this bit indicates that an overflow condition has occurred for the Receive FIFO: The data in the Receive FIFO could not be transferred fast enough to a system buffer in host memory (or to another PCI agent).

5.2.2.4. RxSyncInt – Synchronisation-Error Interrupt

When set to '1', this bit indicates that a synchronisation error has been detected in the packet-synchronising logic on the DTA-120.

5.2.2.5. RxAsiCD – DVB/ASI Carrier Detect

When set to '1', this bit indicates that a carrier signal has been detected on the DVB/ASI Transport-Stream input, and that the PLL (Phase-Locked Loop) monitoring the incoming signal is in lock.

If field `AsiCD` has value '1', this does not necessarily mean that the input signal is DVB/ASI compliant. In particular, connecting an SMPTE SDI source to the DTA-120 will also set `AsiCD` to '1'.

5.2.2.6. RxAsiLock – Locked to DVB/ASI Input

This 1-bit status field indicates whether the PLL (Phase-Locked Loop) monitoring the incoming DVB/ASI signal is in lock.

Value	Mode	Definition
0	NoLock	PLL cannot lock to DVB/ASI input
1	InLock	PLL is locked to DVB/ASI input

5.2.2.7. RxRateOk – Input Rate Ok

When set to '1', this bit indicates that the input rate of the incoming DVB/ASI signal is sufficiently high for further processing on the DTA-120. This bit becomes '0' if the input rate falls below approximately 900 bits per second.

5.2.2.8. RxAsiInv – Invert DVB/ASI Signal – Status

This 1-bit status field indicates whether the DVB/ASI input signal is currently being inverted.

Table 21. AsiInv – Values

Value	Mode	Definition
0	Normal	DVB/ASI signal is <u>not</u> inverted
1	Invert	DVB/ASI signal is being inverted

When the Invert-DVB/ASI-Input Control field in the Receive-Control register is set to **Auto**, then the AsiInv status bit shows the output of the automatic DVB/ASI polarity detector on the DTA-120.

When the Invert-DVB/ASI-Input Control field in the Receive-Control register is set to **Normal** or **Invert**, then the AsiInv status bit is a direct copy of the value of the control field.

5.2.3. Receive FIFO Load (38h)

The FIFO-Load register contains the current load of the DTA-120's Receive FIFO, expressed in number of bytes. Table 22 below is valid for an SDRAM size of 8 Mbytes.

Table 22. FIFO-Load Register – Format¹³

Bit	Mnem	Definition
23...0	FifoLoad	Current FIFO load.
Bits 31...24 of this register are tied to '0'.		

While the DTA-120 is streaming data, the value read from this register is volatile. The value may change with every transmitted byte and with every DMA transfer.

Note

- The actual number of bytes buffered on the PCI card may be slightly higher than FIFO Load due to words residing in pipeline registers.

The maximum value of the FIFO-Load register is (approximately) the size of the Receive FIFO, which is the SDRAM size plus 960 bytes.

The use of the FIFO-Load register in flow-control algorithms is optional. It can be used for enhancing robustness by checking at spe-

cific moments in time whether the FIFO Load is contained within a certain expected range.

5.2.4. Receive Diagnostics (3Ch)

The Receive Diagnostics register contains a number of special fields that can be used for validation and testing of the DTA-120. In normal operation, this register should not be touched. It is recommended to clear the Diagnostics register to all zeros in the device-driver's initialisation routine¹⁴.

Table 23. Diagnostics Register – Format

Bit	Mnem	Definition
7...0	<i>reserved</i>	
8	LoopBack	Loop-back mode
9	<i>reserved</i>	
10	RfRateOvf	Receive FIFO Rate Overflow
11	SdFull	SDRAM Full
31...12	<i>reserved</i>	

5.2.4.1. LoopBack – Loop-Back Mode

Writing a '1' to this bit disconnects the DVB/ASI input circuitry from the Receive FIFO. This enables software to write a test pattern to the Receive FIFO through the Loop-Back-Data register (LbData).

In normal operation, this field should be set to '0'. Loop-Back Mode can be used in the manufacturing test to check data-path and memory integrity.

The Loop-Back-Mode field is not cleared by a software reset.

5.2.4.2. RfRateOvf – Receive-FIFO Rate Overflow

The RfRateOvf flag indicates whether the rate at which data bytes enter the system exceeds the maximum write rate for the SDRAM.

In principle this overflow condition cannot occur on the DTA-120, because the maximum

¹³ Assuming an SDRAM-size of 8 MB. If the SDRAM is larger, more significant bits are included.

¹⁴ Issuing a soft reset through the General-Control register will also clear the Diagnostics register.

write rate to the SDRAM is sufficiently high for a DVB/ASI stream of any rate.

5.2.4.3. SdFull – SDRAM Full

The SdFull flag indicates whether the SDRAM on-board the DTA-120 is full. If this flag is set, further writing to the Receive FIFO is inhibited and DVB/ASI input data will be dropped.

5.2.5. Receive Loop-Back Data (40h)

The Loop-Back Data register can be used to write 8-bit test data to the Receive FIFO in loop-back mode.

Bit	Mnem	Definition
7...0	SfDataNxt	Write data to Receive FIFO
Bits 31...8 of this register are tied to '0'.		

The purpose of this register is to enable diagnostics data-path-integrity and memory-test software.

Note

- Loop-Back Mode (in the Diagnostics Register) must be '1' for meaningful use of Loop-Back Data.
If Loop-Back Mode is '0', writes to the Loop-Back-Data register have no effect.

5.2.6. Receive Valid Count (50h)

The Valid-Count register contains a sample of a free-running counter that is incremented for every "valid" (non-stuffing) byte received at the DVB/ASI input.

Bit	Mnem	Definition
31...0	ValidCnt	Sample of valid-byte counter.

The Valid-Count register can be used to estimate the transport rate of the incoming DVB/ASI Transport Stream.

The Valid Counter is sampled halfway between periodic interrupts (§5.1.2.1): the time interval between two samples is 2^{21} clock cycles of the on-board 40-MHz reference clock (52.4 ms).

The register is designed to be read just after each periodic interrupt, e.g. in an interrupt-service routine.

Note

- The absolute value of the Valid-Count register has no significance. Just the difference between two successive samples is relevant.

5.2.7. Receive Violation Count (54h)

The Violation-Count register contains a sample of a free-running counter that is incremented for every code-violation detected in the DVB/ASI input signal.

Bit	Mnem	Definition
31...0	ViolCnt	Sample of code-violation counter.

A code violation is a bit error that leads to an illegal 8B/10B code (the line code used by DVB/ASI). Bit errors may be caused by poor cable quality, or if the input cable is too long.

Note

- Connecting or disconnecting the cable causes a massive amount of code violations. This is "normal" behaviour, caused by the locking process of the DVB/ASI input chip on the DTA-120.

5.2.8. Receive-FIFO Data (60h...7Ch)

The FIFO-Data register is connected to the output side of the DTA-120's Receive FIFO. This is the main register for receiving MPEG-2 data with the DTA-120.

6. Vital Product Data

Vital Product Data (VPD) is information stored in a PCI device to uniquely identify the hardware and, potentially, software elements of the device. *PCI Local Bus Specification Rev2.2* defines both a standard storage structure and access method for VPD.

The DTA-120 uses VPD to store the serial number, revision level, etc. The sections below list all supported fields. The VPD is stored in the serial EEPROM on-board of the DTA-120. The VPD can be accessed through the VPD-function support built in the PCI-9054.

The DEKTEC DTA-series of PCI cards share the same layout of the serial EEPROM, so that the VPD data can be accessed in a uniform way for each board.

6.1. Serial EEPROM Lay-Out

Figure 12 below shows the memory map of the serial EEPROM and the positioning of VPD elements within the EEPROM memory. Note that addresses are *byte* addresses, whereas the PCI-9054 specification sometimes uses word (16-bit) or long word (32-bit) addresses.

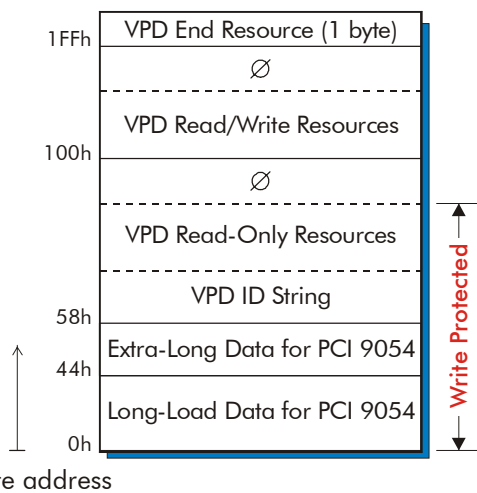


Figure 12. Memory Map of the Serial EEPROM.

The 88 bytes at address 0h...57h are loaded into the PCI-9054 upon power-on reset, to define the initial value of the register set. This part of the EEPROM data is not related to VPD.

The other 424 bytes in the serial EEPROM, starting at address 58h, are dedicated to VPD.

First, the VPD Identification String and the VPD Read-Only Resources are stored sequentially, followed by zero-byte stuffing. The VPD Read/Write Resources are located at address 100h, also followed by zero-byte stuffing. The EEPROM is “closed” by a VPD End resource at address 1FFh.

The VPD Read-Only and Read/Write Resources each consist of a two-character key (e.g. “PN” for Part Number) and a length-prefixed string defining the actual resource.

The first part of the serial EEPROM (register-data for PCI-9054, VPD ID String and VPD Read-Only Resources) is programmed in the factory and write-protected to avoid accidental modification. The VPD Read/Write Resources are not write-protected and may be modified by the end user, e.g. for storing the customer’s system-asset identifier. DEKTEC utilises VPD Read/Write Resources to store software licenses.

6.2. VPD ID String

The VPD ID-String Resource contains the name of the board in ASCII characters. The content of this resource is fixed for all incarnations of the DTA-120.

Addr ¹⁵	Value	Item
58h	82h	ID-String tag.
59h	21h	Length LSB.
5Ah	00h	Length MSB.
5Bh	“DTA-120 DVB/ASI Input 0..214 Mbps”	ID-String data ¹⁶ .

6.3. VPD Read-Only Resources

The VPD Read-Only Resources describe permanent hardware characteristics of the PCI card. This VPD section is stored in the EEPROM just behind the VPD ID-String section.

¹⁵ Byte address in serial EEPROM.

¹⁶ No trailing zero ('\0') character!

The following VPD read-only resources are supported on the DTA-120:

- **PN = Part Number**
The PN Resource is fixed to "DTA-120".
- **EC = Engineering Change Level**
The EC Resource identifies the hardware revision of the board, e.g. "Rev 1".
- **MN = Manufacture ID**
The MN Resource is a 2-digit code¹⁷ identifying the manufacturer of the board.
- **CL = Customer ID**
The CL Resource is a 6-digit code¹⁷ identifying the (initial) customer of the board.
- **SN = Serial Number**
The SN Resource holds a unique serial number. For the DTA-120, this number begins with "4120", followed by a sequence number in 6 or more digits.
- **GC = Guard Code**
The GC Resource is a coded string used in DEKTEC license management.
- **PD = Production Date**
The PD Resource keeps the production date of this DTA-120 instance, e.g. "2002.06".
- **XT = Crystal Accuracy**
The XT Resource lists the accuracy of the 40-MHz crystal oscillator as a string, e.g. "100ppm".

Table 28 below shows an example of the contents of the VPD Read-Only Resources section. DEKTEC reserves the right to append private descriptors in the reserved part of the read-only area.

Addr	Value	Item
07Ch	90h	VPD-R tag.
07Dh	81h	Length LSB.
07Eh	00h	Length MSB.
07Fh	"PN"	VPD keyword.
081h	7	Field length.
082h	"DTA-120"	Part number.

¹⁷ DEKTEC internal code.

Table 28. VPD Read-Only Resources – Syntax

Addr	Value	Item
089h	"EC"	VPD keyword.
08Bh	5	Field length.
08Ch	"Rev 1"	Engineering-Change level.
091h	"MN"	VPD keyword.
093h	2	Field length.
094h	"03"	Manufacture ID.
096h	"SN"	VPD keyword.
098h	10	Field length.
099h	"4120000005"	Serial number.
0A3h	"CL"	VPD keyword.
0A5h	6	Field length.
0A6h	"300004"	Customer ID
0ACh	"GC"	VPD keyword.
0AEh	10	Field length.
0AFh	"TZ)/*L*QNZ"	Guard Code
0B9h	"PD"	VPD keyword.
0BBh	7	Field length.
0BCh	"2002.06"	Production Date.
0C3h	"XT"	VPD keyword.
0C5h	14	Field length.
0C6h	"5ppm@25C;15 ppm"	Crystal Accuracy.
0D4h	"RV"	VPD keyword.
0D6h	29h	Field length.
0D7h	CCh	Checksum.
0D8h	40 x 00h	Reserved.
100h		Read-Write section

The length of the VPD Read-Only Resources section is tuned such that the VPD Read/Write Resources section starts at byte address 100h.

6.4. VPD Read-Write Resources

The VPD read/write section can hold 255 data bytes that can be updated dynamically from software. Potential usage includes diverse applications such as software keying, system-asset identification and storage of fault codes for inspection by service personnel.

Every byte in the serial EEPROM can be rewritten about 10⁶ times. Therefore, the VPD Read/Write Resource cannot be used for data that is updated a lot, e.g. every second. It is recommended to use the Read/Write section only for data that has a near-static nature.

The following standard tags are defined in *PCI Local Bus Specification Rev2.2*.

- **Vx = Vendor Specific**
This is a DEKTEC-specific item, e.g. a software license. The second character (x) of the keyword can be 0 through 9 and A through Z.
- **Yx = System Specific**
This is a system-specific item. The second character (x) of the keyword can be 0 through 9 and B through Z.
- **YA = Asset Tag Identifier**
The resource contains the system-asset identifier provided by the system owner.
- **RW = Remaining Read/Write Area**
This descriptor is used to identify the unused portion of the read/write space.

The data bytes are stored in the serial EEPROM at address 100h up to 1FEh inclusive. The byte at address 1FFh is used to store the VPD-End tag. Table 29 below shows an example of the syntax of the VPD-Read/Write-Resources section.

Addr	Value	Item
100h	91h	VPD-W tag.
101h	FCh	Length LSB.
122h	00h	Length MSB.
103h	"V3"	VPD keyword.
105h	16	Field length.
106h	"3rS=;2kl`MD(#ac"	License string.
116h	"YA"	VPD keyword.
118h	25	Field length.
119h	"DVB/ASI Test Analyser #11"	System-asset identifier.

Addr	Value	Item
132h	"RW"	VPD keyword.
134h	202	Field length.
135h	202 x 00h	Reserved.
1FFh	78h	VPD End tag.

6.5. Reading VPD Data

VPD Resources can be read 4 bytes at a time with the procedure described below. The hardware does not support any form of parsing VPD data, this is the job of the device driver.

1. Ensure that the read address is 32-bit aligned: the two least-significant address bits shall be zero.
2. Write the address to the 16-bit PCI-9054 register **PVPDAD** at PCI offset 4Eh in PCI-Configuration Space.
Set the **VPD-Address** field to the read-address field (last two bits 0) and, in the same operation, set the **F-flag** field to '0', signalling a read operation.
3. Poll the **F-flag** in a loop until it becomes '1'. This indicates that the VPD read data is actually available.
4. Read the 32-bit PCI-9054 register **VPDDATA** at PCI offset 50h to obtain the requested 4 VPD data bytes.
5. Repeat steps 1..4 for all VPD words to be read.

6.6. Writing VPD Data

The VPD Read/Write Resources section can be written 4 bytes at a time with the procedure described below.

1. Ensure that the write address is 32-bit aligned: the two least-significant bits shall be zero.
2. Enable programming of the serial EEPROM by writing a '1' to PE in the General Control register (§5.1.1).

3. Change the *Serial EEPROM Write-Protected Address Boundary* register in the PCI 9054 (register `PROT_AREA` at PCI-offset `0Eh`¹⁸) to a value less or equal than the write address divided by four.
The division by four is required because `PROT_AREA` contains a 7-bit field that points to a 32-bit "long-word" address.
4. Write the desired data (32-bits!) to PCI-9054 register `VPDDATA`.
5. Write the destination address to the 16-bit PCI-9054 register `PVPDAD` at PCI offset `4Eh` in *PCI-Configuration Space*.
Set the **VPD-Address** field to the write address (last two bits 0) and, in the same operation, set the **F-flag** to '1', which signals a write operation.
6. Poll the **F-flag** until it changes to '0' to ensure that the write operation has completed.
7. Repeat steps 1..4 for all VPD words to be written.
8. For safety, change `PROT_AREA` back to `7Fh`, and:
9. Disable programming of the serial EEPROM by writing a '0' to `PE` in the General Control register.

Note

- It is the responsibility of the device driver to maintain integrity of the VPD Resources.
For example, if a VPD Resource to be rewritten does not start at a 32-bit boundary, then the host should first read the original 32-bit VPD word, AND/OR-in the new data, and write the resulting 32-bit word back.

¹⁸ In PCI-memory space.