An Implementation of 100Mbps TAXI Subscriber Board

Chan Kim, Jung Hoon Paik
Electronics and Telecommunications Research Institute
161 Kajeong-dong, Taejon, Korea

Abstract

There have been proposed and standardized several efficient physical layer interfaces by the ATM Forum. Among these standards, 100Mbps TAXI is the second most popular in the ATM market. In this paper, an implementaion of 100Mbps TAXI subscriber interface board(TSIB) for a B-NT2 system called CANS is described and some design considerations are addressed. The board was designed and implemented simply and can transmit and receive data at cell rates of up to 98.148Mbps which is the maximum data rate of the specification due to the TT overhead. In addition to this basic cell transferring function, various mechanisms were adopted to meet the system requirements and to make the circuit more fault-tolerant.

1. Introduction

There have been proposed and standardized various types of interfaces for B-ISDN UNI which can be implemented easily and economically. Some of them are still being discussed in the ATM Forum. This paper describes a 100Mbps TAXI subscriber interface board(TSIB) designed to be used in a B-NT2 system called CANS(Centrallized Access Node System) which is now in its final stage of the developement. This board follows the "100Mbps multimode fiber interface specification" of the ATM Forum UNI 3.0 and will be used as a subscriber interface board in CANS along with 155Mbps STM-1 subscriber board and 25.6Mbps UTP subscriber board. This board has two ports and could be updated to serve four subscribers if it is required.

2. 100Mbps TAXI Interface Specification[1]

In this section we briefly look at the ATM Forum's 100Mbps TAXI interface specification. This interface has several functions shown and described below.

a. Optical Interface

Converts electrical signal to light and vice versa. The optical interface follows FDDI PMD specification. To name a few of it, it uses 62.5/125 multimode fiber and 1300nm light and the link can be up to 2kms long.

b. 4B/5B line coding

Line coding follows ANSI X3T9.5's 4B/5B coding. In 4B/5B coding each 8-bit data is divided into 4 bits and each 4 bits are coded into 5 bit codeword. Of the 32 patterns possible with 5 bit coding, only 16 patterns are used as coded data and the other 16 patterns are used as control symbol. Two control symbols are combined together to form a command. In this specification, only three were defined now. They are JK which is the Idle pattern (or Synch pattern) and TT which indicates Start of Cell and QQ which is used to inform the counterpart entity which is connected through optical fiber that LOS(Loss Of Signal) has happened in the receiver. The use of these commands will be explaned later in this section.

The data rate of this interface is 100Mbps but since 4B/5B coding is used, the line rate is 125Mbaud.

c. HEC generation/verification

The transmitter calcultes the HEC using the first 4 bytes of the cell which is being transmitted and inserts the HEC value into the cell's fifth byte location. The HEC is then recalculated at the receiver and compared to the received value. If the HEC is correct, the cell is received but if not, the cell is discarded. The HEC is calculated in the same way as is specified in ITU's I.432 but no header correction is performed, which is different from that of I.432.

d. Cell Delineation

Cells are transmitted asynchronously on the line and when cells are not transmitted, JK synch pattern is transmitted to help the receiver recover the clock and do the byte alignment. Just before a cell is transmitted a special code TT(Start of Cell) is transmitted so that the receiver can know when the cell is comming. That is to say, 54 bytes(TT + 53 byte long cell) are transmitted contiguously on the line and cells can be transmitted back-to-back. Due to the TT overhead, the maximum

cell transfer rate is 100x53/54 = 98.148Mbps.

3. Design of 100Mbps TAXI board

3.1. Interaction with other boards in CANS system

Figure. 1 shows the relation of TSIB with other boards in CANS, the B-NT2 system which is developed in ETRI. The CANS system consists of two CPU boards and two ATM processing boards which does the cell header processing and routing, and physical layer boards for network and subscribers. The physical layer board for the network side is of type 155Mbps STM-1. The CANS have three different subscriber interface boards for different subscriber types. They are for 155Mbps STM-1, 100Mbps TAXI, 25.6Mbps UTP interfaces. These physical layer boards have the same interface for the upper ATM board and share the same slots in the CANS system. The subscriber boards are hot swappable and the ACPB automatically reinitializes the board and reports this change of configurations to system management block. A CPU board called ACPB(ATM Control Processing Board) directly controls the function of the board including many real-time processing. There is another CPU board which does higher layer functions such as signalling, OAM, network management, etc. The two CPU boards communicate with each other through IPC(Inter-Processor Communication) using DPRAM. TSIB is one of the subscriber boards that was designed for 100Mbps TAXI interface. and like other subscriber boards, TSIB receives cells from ATMB(ATM Layer Processing Board) and transmittes them onto the optical fiber and vice versa. The timing diagram of cell transmission/reception between ATMB and TSIB is shown in figure 2. The TCA goes high when the transmitter buffer has at least one cell space empty and can receive a cell from upper ATM layer. The RCA means that the receiver has received at least one cell and it is(or they are) in the receiver FIFO waiting to be read. The TSOC,RSOC indicates the start of cell position.

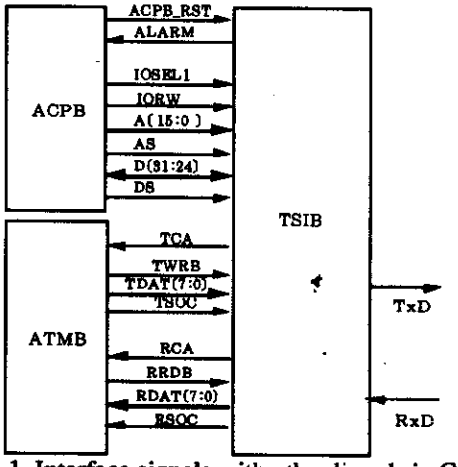


Figure 1. Interface signals with other boards in CANS

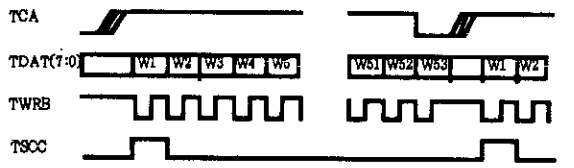


Figure 2a. Transmitter interface timing diagram with ATMB

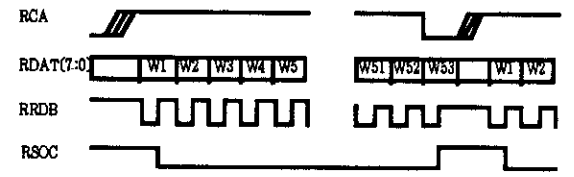


Figure 2b. Receiver Interface timing diagram with ATMB

3.2. Hardware architecture[2],[3]

Figure 3 shows the hardware architecture of TSIB. In this figure TAXI-TX,TAXI-RX chips are AMD's AM7968, 7969 respectively and Sumitomo's SDM3201-XB was used as the optical transceiver. FIFO chips are IDT's IDT7200. Almost all logics were implemented in two Actel 1225 FPGAs (Each for transmitter and receiver) except FIFO, TAXI, and Optical trasceiver and performance monitoring error counters.

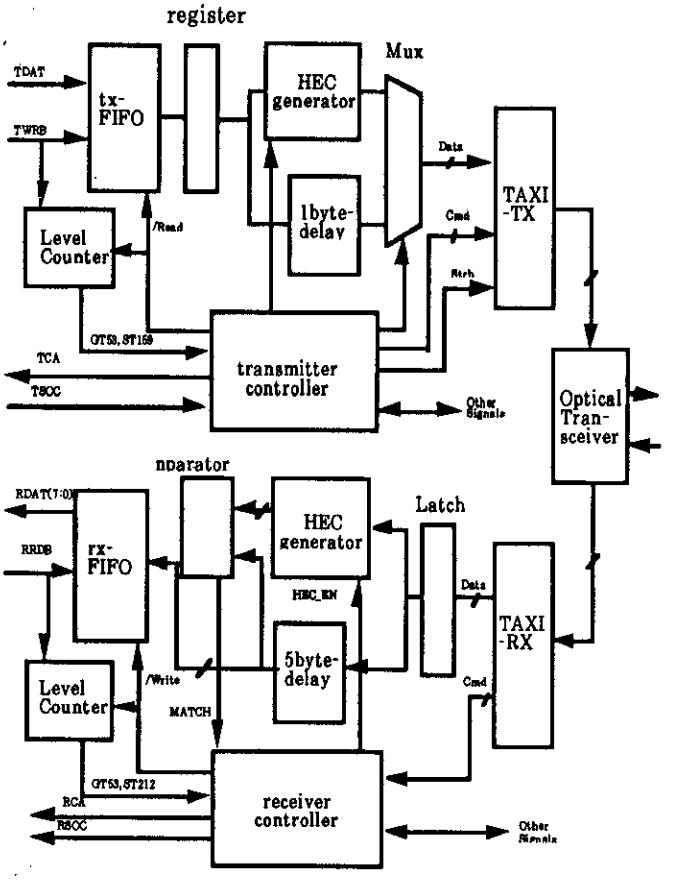
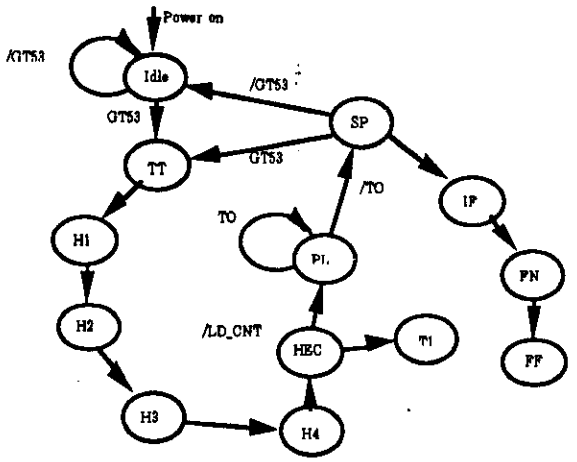


Figure 3. Hardware Block Diagram of TSIB

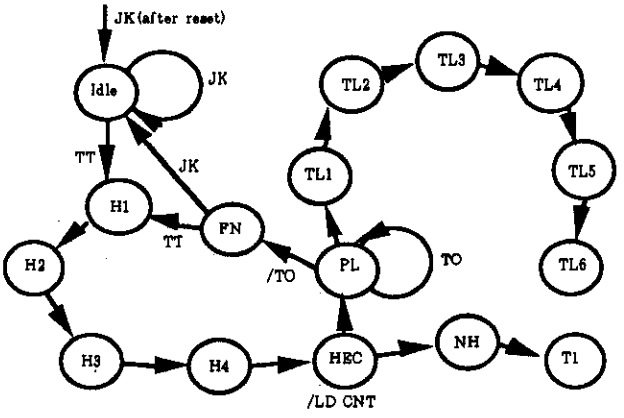
First, the transmitter section is described. The level counter keeps the level of data in Tx-FIFO using the FIFO's write and read signals and generates GT53(greater than 53), ST159(smaller than 159) signals. The TCA generator receives the ST159 signal and generates TCA. When there are cells to transmit and TCA is high, the ATMB sends a cell to TSIB. When level counter informs the controller of at least one cell's waiting in FIFO, the transmission routine is started. The controller is a one-hot statemachine which is in idle state during its waiting period and starts when GT53 is high and with the transition of states, gererates signals to control the data path, and then returns to idle state if there's no more cells waiting in FIFO(GT53 low). When there are more cells to be transmitted after one cell transmission, the controller jumps to a second state so that cells can be transmitted without loss of clock periods and this is to obtain the full bandwidth of 98.148Mbps in case of full service. During the transmission routine the cell data is read from the FIFO and sent to TAXI-TX and the fifth HEC field is replaced with the HEC calculted by the HEC generator. The HEC generator is a parallel version of the serial CRC circuit and it is a well known technique to convert a serial circuit to parallel one. Before the first cell data is strobed into TAXI-TX, the TT command is strobed so that TT can be inserted before the head of a cell. The TAXI-TX does the remaining task of coding data or command using 4B/5B rule, inserting JK when no data is being strobed, and serializing the data out to the transceiver. The transceiver converts the PECL differential signal to optical signal and trapsmittes light out onto the optical fiber. Figure 4 shows the transmitter's state diagram. A down counter was placed in the controller to control the length of transmission routine.



HEC_CLR = TT
HEC_EN = H3+H4+HEC+T1
HEC_SEL = T1
RD_EN = H1+H2+H3+H4+HEC+PL
STRB_EN = H2+H3+H4+HEC+PL+SP+IF+FN

Figure 4. Transmitter's State Diagram

The function of the receiver is as follows. The recevied light is converted to PECL electrial signal and sent to TAXI-RX chip. TAXI-RX chip deserializes the data into bytes using its byte alignment and 4B/5B decoding, and strobes the data / command to receiver logic. Receiver can be divided into datapath and controller. The controller is also a one-hot statemachine and it waits in idle state when no cell is being received and when TT is detected it starts its routine of cell reception and transitioning through states, generates control signals. The received bytes are sent to 5-byte shift register and at the same time to the HEC generator. After comparing the generated HEC value with that of the cell being received, the result is stored in a flip/flop so that it can be used to enable or disable the following rx-FIFO write signal. When a cell is written into the rx-FIFO, the level counter detectes that one cell is in rx-FIFO and informs the controller of the fact. The RCA generator generates RCA so that ATMB can know there is at least one cell to be read. To prevent the case when the ATMB neglects for any reason its duty of reading cells as fast as possible and rx-FIFO gets into the disastrous overflow state, the whole one-cell-write into rx-FIFO is disabled when there are 4 cells (this information comes from level counter ST212 signal). For the case when cells are coming contiguously on the line, the receiver controller checks in a proper period if a new cell is coming before its completion of one cell writing and, if necessary, jumps to an appropriate state to receive the new cell while writing the final bytes of the preceding cell. In this way, though the cell write is delayed more than 5 clock cycles due to the HEC verification, no cell is lost when cells are coming back-to-back. Figure 5 shows the receiver controller's state diagram.



HEC_CLR = IDLE&TT
HEC_EN = H1+H2+H3+H4
TRAIL = TL1+TL2+TL3+TL4+TL5+TL6
WR_EN = (HEC_OK+IGN_HEC)&TRAIL
HEC_OK = NH

Figure 5. Receiver's State Diagram

When RCA is high, ATMB reads the cell from the rx-FIFO. The ST159, GT53 cannot be used directly to make TCA,RCA but TCA, RCA generator is needed to make the signal go low according to the timging shown in figure 2.

Figure 6 shows the timing of the transmitter and receiver logic between FIFO and TAXI chips. The transmitter reads data from tx-FIFO and overwirtes HEC in its position and inserts TT before sending a cell . When TT is detected and after checking if HEC is correct the receiver writes cell data to rx-FIFO. In datapaths the registers were placed appropriately in a way that the controllers work closely with the datapath to do the task.

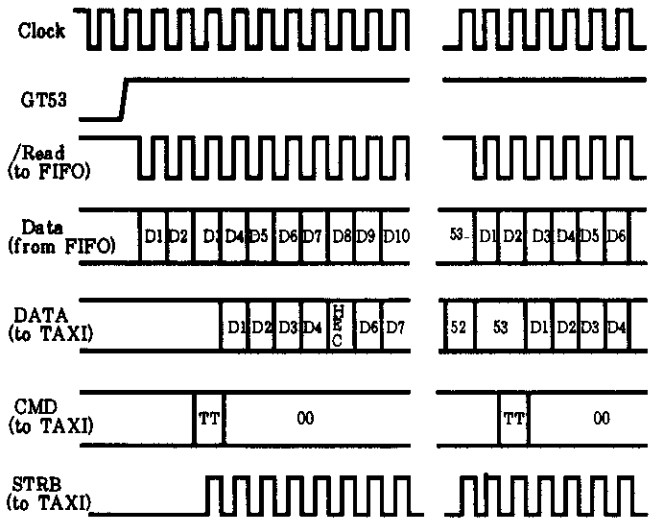


Figure 6a. Interface Timing Diagram between FIFO and TAXI-TX

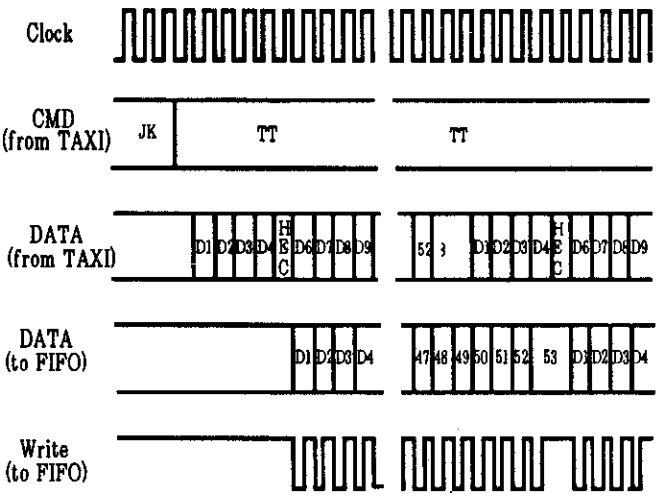


Figure 6b. Interface Timing Diagram between FIFO and TAXI-RX

In addition to these main transmitter and receiver logic, there are QQ sending logic which periodically sends QQ when the receiver is in LOS(Loss Of Signal) state. There are also counters in the receiver to count the number of HEC errored cells(discarded cells) and number of 4B/5B violations. These counters happened to be of size 15 but they are saturation counters which do not overflow but stay at 15 and resets after the CPU reads the value.

3.3. Software Architecture

Like most physical layer boards, most important functions are implemented through hardware but there are some tasks to be done with software. This board goes to normal operating mode after power-on reset but for general case's initialization(when the board was inserted into an traffic-loaded slot, or when the operator requested a port to be initialized) and OAM purpose(the change of the line status) and for performance meauring some software operation is neccessary. When interrupt is generated by hardware, interrupt routine is called by main controll program, and in the ISR(Interrupt Service Routine), the source of interrupt is found out and appropriate actions are taken.

Figure 7 shows the interaction of TAXI software with other software blocks. Since the physical layer can have three different types of interfaces, the physical layer software is divided into three sub-blocks which are SLCB(STM physical Layer Control Block), TLCB(TAXI physical Layer Control Block) and ULCB(UTP physical Layer Control Block). In addition to these blocks, there is one block which controls these three blocks as the physical layer master block. These block is called PLMB(Physical Layer Master Block) and it is placed between these three physical layer blocks and upper layer software. It provides a unified view of physical layer to the upper ATM or OAM block. The arbitration block calls the specific initialization routine when the system is initialized or when a specific port's initialization request packet was received from the system management block through IPC(Inter Processor Communication) or when a new board was inserted. In each case the arbitration block identifies the board type by reading specific addresses. When alarm is generated on specific port, the PLMB calls the corresponding alarm routine according to the board type of the port. The PLMB calls corresponding performance routine every second and every 15 minuite and when the performance data was requested for a specific port from the management through IPC. In these cases the calling occasion is given by separate "case" argument.

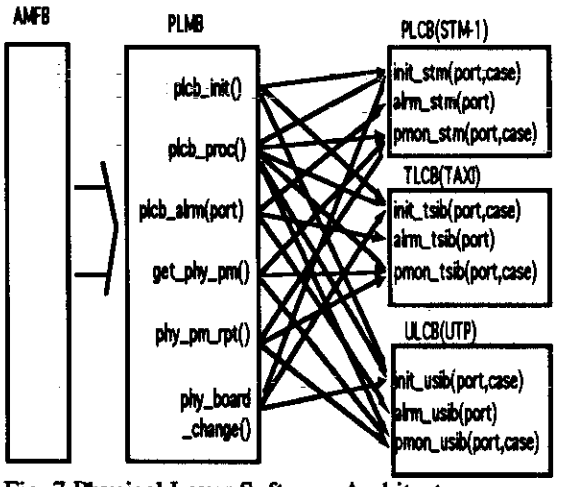


Fig. 7 Physical Layer Software Architecture

Like other physical layer sub-blocks, and as explained before, the TLCB consists of three routines. The first one is the initialization routine which is called by the PLMB with a port argument when the port is of type TAXI. This initialization routine for a port is called when the whole system is initialized and when the system management requested a specific port to be initialized or when the board was newly put into a slot during the system operation. In each case the routine initializes the port by resetting the board and after reading the status, it reports the result by returning a value or sending an IPC message according to the case. When reading the status some time delay should be given before reading the status register after reset since the physical layer circuit(especially the PLL) needs time to stabilize after reset.

The second routine is alarm processing routine which is called when alarm was generated on a port and its type is TAXI board. In TAXI board, the receiver interrupt is generated in the following event and according to the event, the follwoing actions are taken. When the light signal goes off or on(this happens when the line is disconnected or connected), the change is reported by IPC to the system management block. The TAXI software keeps the signal status because the status register only shows the current signal status. This is to know whether it is going-on or going-off. When the rx-FIFO becomes full, the FIFO is reset. When QQ was received, the routine sends an IPC message to the management for a message display so that operator may see the message and do some checks. Transmitter interrupt is generated when tx-FIFO becomes full and in that case the tx-FIFO is reset. The hardware is devised not to lose the cell alignment in case of FIFO reset.

The third routine is performance routine. When it is called by every-1-second base, it reads the error counter and adds the value to corresponding cumulation variables. The board has two error counters which are HEC error count and 4B5B error count. When the routine is called by every-15-minuite base, the routine sends the cumulated performance data to the OAM block through IPC message. When it is called due to the IPC request for performance data, it sends the data to the management through IPC.

Table. 1 shows the command/status register of the board arranged and implemented into the hardware to control the operation. The identification bits were added for the board type identification purpose. In the identification, there should be both 1 and 0 to be able to tell that the board exists. This is because some buffers take the input for 0 when it is not driven and others take the input for 1(like TTL input which is pulled up internally) .

Table 1. Command/Status Register of TAXI subscriber board

	COMMAND	meaning	STATUS	meaning
7	SW_RESET	S/W reset	SW_RESET	in S/W reset state
6	Not Used		1	used for identification
5	Tx INT CLR	clear transmitter interrupt	Tx FIFO FF	tx-FIFO is in full state
4	Tx_FIFO_CLR	clear tx-FIFO	0	used for identification
3	Not Used		SIG DET	no optical signal recevied
2	Block_QQ_INT	disalbe QQ_Rcvd interrupt	SYNC	receiver out of synchronization
1	Rx_INT_CLR	clear receiver interrupt	Rx_FIFO_FF	rx-FIFO is in full state
0	Rx_FIFO_CLR	clear rx-FIFO	QQ_RCVD	QQ code has been received

4. Some Practical Design Considerations

In this section, some design considerations in designing this board are addressed which can cause problems if not considered. These kinds of techniques are well known to many experienced designers but are briefly discussed below to be of help for future designers.

a. Generating FIFO's read or write signal

In most high-speed design, the design efforts are put into the retiming of the logic. If the combinational logic delay is greater than the clock period, retiming is neccessary(that is, cut the combinational path with flip-flop and adjust related timings)[4]. When generating asynchronous FIFO's read or write signal by gating the control signal and clock, the control signal's delay from clock edge can make problems. Figure 8 shows this case. In the figure, assume the /Enable signal was generated by decoding some states. Then the decoded signal is delayed from the clock edge by the decoding logic especially when two or more stages were used for decoding the signal. In this case, as shown in the timing diagram, the first write pulse has shorter width

than others and after the final write pulse there arises glitch. Of course delaying the write clock can help. but the correct way is to retime the control signal so that it is no longer a delayed signal. In figure 8, the /En1 and /En2 signals are functioally the same but taking into account the combinational delay, /En2 is faster than /En1 by the propagation delay of the AND gate. These kinds of circuit conversion need to be performed to make a control signal faster to prevent the races or glitchs. There is one rule, if a fast control signal which is not delayed from the clock edge is needed, decode the signal one or more clocks earlier and use the clocked version of it.

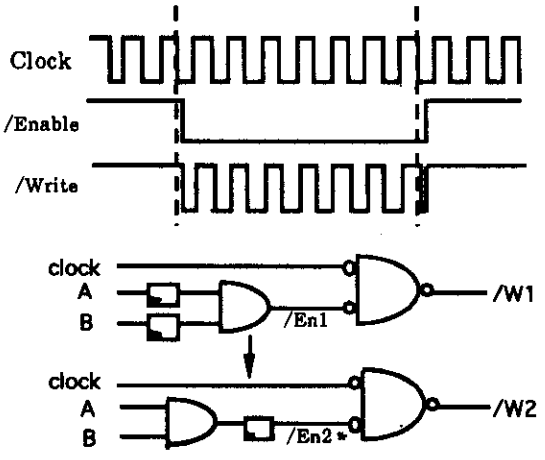


Figure 8. Example of Retiming to reduce a signal delay

b. Reading asynchronous FIFO

When the FIFO doesn't have latch, the read data goes away immediately after the end of /read signal so we need to latch the data with the rising edge of the /read signal. But when the reading rate is fast, care must be taken not to lose a data. The problem arises when the data from FIFO cannot come to the latch input by the rising edge of the /read signal. If FPGA is used, as figure 9 shows, the /read signal is delayed due to the output buffer of FPGA and data from FIFO is delayed due to the access time plus input buffer delay so that by the time /read signal goes up to latch the data inside the FPGA, the data has not yet come to the D latch input. In this design, the total sum of the delay exceeds 40ns which is the pulse width of the read signal(clock periode is 1/12.5MHz = 80ns). Therefore the data should not be latched by the read signal but should be clocked in by the next clock since the data will be valid for 40ns no matter how much it is delayed from the clock(The total sum of the delay does not exceed 80ns though).

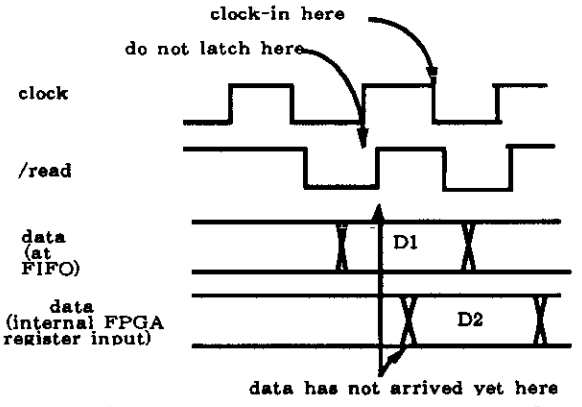


Figure 9 Timing of reading latched output FIFO

There is another reason that the FIFO's data must be first clocked into a register. As shown in the hardware block diagram(Figure 3), in the transmitter, a register was placed between FIFO and the HEC generator for synchronization purpose because the HEC calculation needs relatively long time due to the multi-stage XOR gates. If the FIFO's output were to be used directly for HEC calculation, the HEC generator's XOR gates would fall short of time to make the next values before the clock arrives because the data is supplied much later from the active clock edge. Generally speaking, any ashynchronous input must be synchronized by clocking before it is used in the synchronous world.

c. FIFO level counter design

There are several ways of implementing FIFO level counter. We could use an up-down counter arbitrating the read/write signals to count up or down since the up-down counter generally cannot count in both direction. Through arbitration, we should give the counter either a direction signal or two separate up/down pulses which should not overlap according to the scheme of the counter. This method requires fast logic to sample the read/write signals and determine the direction and is not recommended in high-speed design. Another method, using two up-counters and subtracting the read count from write count is not a good choice since most logic libraries and logic families do not provide subtractors but only adders and this makes subtraction expensive(additional 2's complenting, or inverting all bits and adding 1). We can also use an up-counter and a down-counter to count in each direction and add the result to get the level. This is the method used in this design. This method needs some means to remove the decoded signal(GT53,ST159) of glitchs generated by the counter's combinational logic which could lead to a false trigger if not removed. In this design a purely combinational circuit using delay was used to remove the glitchs and it worked fine. Another good approach is to use cell counter in which

the level is counted by cells(of course, two 53 byte counter is needed in each direction).

d. Noise problem

In analog chip, the noise in the power pin directly and adversely affects the signal hence deteriorating the chip's performance. To avoide this situation, the TAXI chip's three power pins which are adjacent to each other were isolated by making ditch between them on the VCC power plane. And the power pins were decoupled with two different valued capacitors each to remove not only normal frequency noise but also relatively higher frequency noise. The optical receiver, which is also sensitive to power noise since it was designed to amplify very weak signal to restore the digital signal, was also taken care of. The optical module's power plane was isolated from other digital part of the board and connected to digital power plane by ferrite bead. But its ground plane is common with other area providing return path for the PECL highspeed signal.

e. Initializing receiver controller statemachine

In designing the receiver controller we should take into account the fact that the TAXI-RX does not settle to its normal operating condition immediately after the reset(when JK is comming and the clock difference is 1%, it takes 7-8 microseconds to lock-up). If we want the controller to go to a known state(which is the idle state in most cases), the jump condition should be JK only. If the condition to jump to idle state after power-up is made conventionally by ANDing the preset F/F and decoded JK, the circuit goes nowhere because the JK pattern cannot be detected in a clock cycle after power up. Therefore, the condition to go to the idle state after power-up must be only JK. Then some clock periods after reset, the circuit goes to idle state and this makes the receiver jump to idle state whenever JK is received.

f. Decoding TT or QQ in the Receiver

In TAXI, TT is coded into "0010" in command. If just this pattern was decoded to activate the receiver logic, it makes problem when the line is disconnected. This is because when the receiver line is disconnected, the random signal from the optical module generates occasional mimic pattern of TT causing the receiver to begin its cell receiving routine. Though the HEC will be incorrect in most such cases, there can be chances of correct HEC with probability of $1/2^8$ which causes garbage data to be passed as cells to ATM layer board. So the optical receiver's LOS(loss of signal) should also be included in the TT decoding so that TT can be true only when there is light. This also applies to QQ. There should be another care to be taken. Right after reset, the TAXI-RX chip, during some settling period, gives off random data which can include QQ(and this is always the case) and generate false QQ-RCVD alarm. In this design, the QQ interrupt was avoided by disabling the QQ interrupt with preset flip/flop during reset and enabling it with software by writing 1 to a command bit.

g. Generating alarm after reset

In our system, the physical layer board was requested to generate the LOS(Loss of Signal) alarm when the physical layer port was initialized with its receiver line plugged out. The TAXI board's signal detect alarm was generated when the two cascading flip/flops' value differ at any moment. Since the status register only gives information on current line state, the software should remember the last value of the signal detect to be able to tell if the change was signal going on or going off. The two flip/flop is preset during reset thus making an alarm after reset by active high "signal detect" signal from the optical modulewhen it is initialized with its receiving line not connected.

h. Clearing FIFO during the operation

Due to its nature, FIFO devices cause problem if one or more bytes of data(not whole sized data like one cell) present in it. If this misalignment happens for any reason, later reading of the data becoms obsolete and further processing is impossble(The hardware runs but with wrong data!)

Since the ATM layer board is allowed to send cells only when it has been signaled through TCA that the physical layer transmitter buffer has space for at least one cell, the tx-FIFO never goes into full state in normal operation. However the problem can arise when the tx-FIFO is cleared during operation, that is when connections have been set up and cells are moving through the interface or when the board is inserted into such active ATM-PHY interface. This is because at the end of FIFO-clear, the ATM layer can be writing cell data or the transmitter can be reading cell data thus pushing/leaving some garbage data into/in the FIFO. When there is no physical layer board, the TCA signal is pulled low by the ATM layer and cells are waiting in the ATM board's buffer. When the buffer goes full, it is cleared. And it repeates this process until the TCA goes high by the physical layer. When the board is inserted into the slot, the FIFO-clear is pulled low for some period by the power-on-reset and then S/W reset by initialization, and the level counter is also cleared by FIFO-clear signal. In this case the TCA should be taken care of not to be in high state during reset though the level counter is zero during that period. This is because the ATM board does not know whether the physical layer is in reset condition but only sees TCA. In this design, the TCA was overwritten by low until after 2 or 3 clocks after FIFO-clear finished. And since the

transmitter's FIFO-clear is the ORed version of reset and S/W FIFO clear command, this FIFO protection applies to H/W, S/W reset and S/W FIFO clear.

For the rx-FIFO reset, similar scheme must be used. When cells are coming in from the line but the ATM layer is busy servicing other ports, the rx-FIFO can go into full state. Then the rx-FIFO should be reset by software. Though automatic hardware reset can be adopted, it was not chosen to provide a way to report to the upper layer at the same time. The cell misalignment happens when at the moment of FIFO-clear signal's going high, the receiver circuit happened to be in the middle of writing a cell into the FIFO or the ATM lyayer happened to be in the middle of reading a cell. To prevent the FIFO from going out of this cell alignment, some protection scheme was devised. First, during the reset, a new start of whole cell writing is blocked by hardware and a new start of whole cell reading is also blocked by the RCA which is in low state by cleared level counter. In addition to this, the software should make sure that when reseting the rx-FIFO with software, the reset signal should be long enough to let any on-going writing or read operation finish. By this means, after FIFO-clear is finished, the receiver circuit starts over from empty FIFO. Again since the receiver's FIFO-clear signal is the ORed version of reset and S/W rx-FIFO clear comand, the same protection is in effect for HW or SW reset and S/W FIFO clear.

5. Conclusion

There have been proposed and standardized several efficient physical layer interface specifications. Among these standards, 100Mbps TAXI is the second most popular in the ATM market. In this paper, the implementaion of 100Mbps TAXI subscriber interface board for a B-NT2 system called CANS was described and some design considerations were addressed. The board was designed at low cost and worked fine at up to 98.148Mbps which is the maximum data rate due to the TT overhead. In addition to this basic cell transfering function, various mechanisms were adopted to meet the system requirements and to make the circuit more fault-tolerant. Due to the simplicity of the specification, the implementation was simple and easy, and the probabilty of the receiver to lose a cell was extremely low(this happens when TT code is corrupted so that the receiver cannot detect the arrival of a cell). And since the JK pattern is transmitted during idle period there was no synchronization problem.

< References >

1. "100Mbps Multimode Fiber Interface Specification," ATM User-Network Interface Specification Ver. 3.0, The ATM Forum, Sep. 1993.
2. *Am7968/Am7969 TAXIchipsTM Handbook*, AMD, Sunnyvale, CA, 1992.
3. *Actel FPGA Databook* , Actel Corp., Sunnyvale, CA, 1994.
4. G.Michelli, "Synchronous Logic Synthesis : Algorithms for Cycle-Time Minimization", IEEE Trans. on CAD. Vol. 10, No. 1, Jan 1991.