NICS–TNG Infrared Camera
A FIFO buffer for the ADC board

G.Comoretto[1], C. Baffa[1], F. Lisi[1]

n. 8/99

October 1999

[1]Osservatorio Astrofisico di Arcetri,
Largo E. Fermi 5, I-50125 Firenze (Italy)

Arcetri Technical Report N° 8/1999

Abstract

For faster acquisition rate with the CCD Controller of the Galileo project, a FIFO buffer is needed after each ADC channel to accomodate for the relatively slow response time of transputer links. This report describes the FIFO/ADC board implemented as part of the NICS camera controller. The FIFO is realized using fast static RAM and a Xilinx chip, which serializes the ADC parallel data on a transputer link. The board is controlled by the DSP of the CCDC. The DSP can also insert arbitrary words in the transputer link, to implement arbitrary protocols.

1 Introduction

The Galileo CCD Controller (CCDC) is based on the INMOS transputer TRAM standard. Data acquired from the CCDC are processed in a network of transputers, and sent to the control workstation using a fiber optic transputer link.

For very fast conversions, the transputer link poses strong timing limitations. Although it has a maximum nominal speed of 9 MB/s, and has a dedicate DMA channel for each link, the time required to program one of the DMA channels can exceed several tens of microseconds. Moreover, when all links are used, task priority issues can further degrade the service time.

A FIFO buffer is therefore required in the data acquisition path to achieve a very fast conversion rate. Such a buffer is also required for fast acquisition of subframes, when the required instantaneous conversion rate is higher than the average rate.

The size of the required FIFO prevents the use of commercial chips, but can be easily implemented using static RAM. The FIFO control logic can be implemented using a FPGA, like a Xilinx chip. Since also the serialization of the parallel ADC output must be realized in custom logic (in a Xilinx chip in the current design), adding the FIFO logic to the existing ADC board design is rather straightforward.

1.1 Document organization

This document is divided into three main sections. This section gives a quick overview of the board structure and functionality. Section 2 describes in detail the of the buffer and its components. Section 3 describes the board, its settings, and the programming model.

1.2 The ADC board

The ADC board is described in full detail in [1]. Here the main board characteristics are summarized. A block schematic of the board is shown in fig. 1

Each ADC board contains two fast ADC's (Analogic 4320, 4322 or 4325), that can convert a signal with an accuracy of 16 true bits with a sampling rate of resp. 1, 2 and 0.5 MHz. The input signal in each channel, in differential form, is first amplified by a Burr Brown INA110 instrumentation amplifier, and then converted at the appropriate sampling rate by the Analogic ADC. The conversion is asyncronous, and is started by a strobe signal from the decoded CCD sequencer bus (port SEQ A).

The two input channels are independent, and have independent, locally regulated, analog power supplies.

The digital samples are processed by one or two transputer (TRAM) modules. They are sent to the transputer(s) as two TRAM serial links. The FIFO chip, described in the next section, both converts the parallel 16-bit samples to two TRAM serial words, and stores the incoming samples, if the transputer is not ready to accept them.

The transputer module(s) are inserted in the TRAM link chain, running inside the backplane of the CCDC bus. If only one module is present (as in the NICS controller) it serves both ADC's, and is connected to the LINK1 chain in the CCDC bus.

The FIFO is implemented as a XC4006-160 programmable gate array and two NEC μPD431008L 128K×8 bit memory chips. Parallel words from the two ADC's are stored as 16-bit samples in the memory, and converted to serial format when the transputer requires them. The chip implements two independent buffers, one for each input channel, with a depth of 64 Kword each.

The chip is also mapped in the CCDC decoded addressable space. The DSP can write in the FIFO buffer, adding protocol words to the TRAM data stream, read the buffer (mainly for debugging purposes), reset the FIFO and read its current status.

Four LEDs, mounted on the board front panel, monitor the FIFO status (full, empty, or used) for each channel.

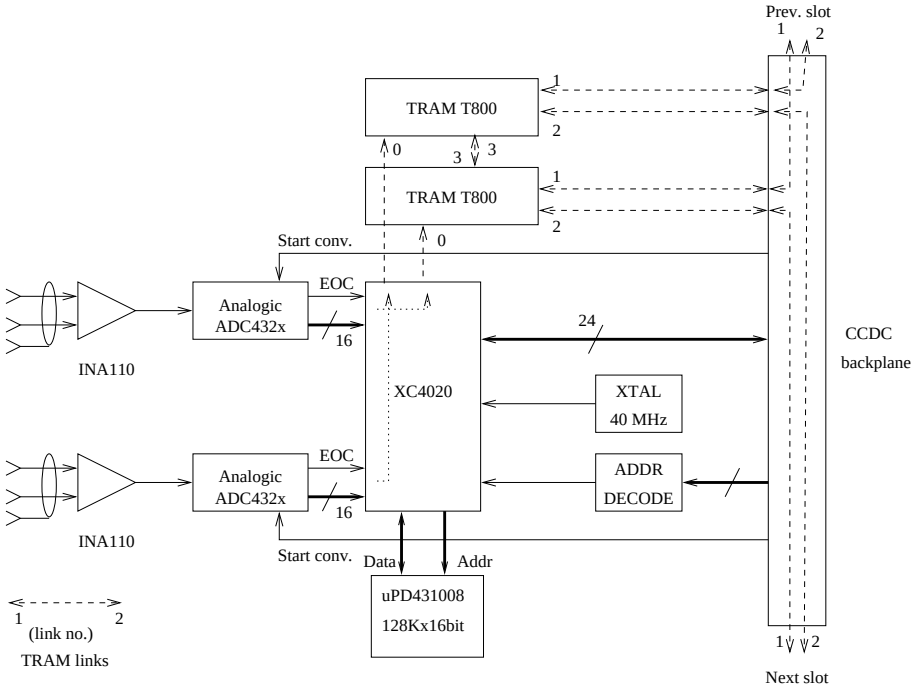

Figure 1: ADC board block diagram

2 FIFO buffer

Each ADC card contains two ADC chips, a Xilinx FPGA that serializes the two data streams according to the INMOS TRAM protocol, and one or two transputer TRAM modules.

Adding a static RAM bank allows a FIFO buffer of arbitrary size to be implemented. The FPGA generates the correct memory read and write addresses, and controls the FIFO operation.

In this architecture, the FPGA is composed of the following subunits, connected by a generic 16-bit wide bus:

- Two input registers, one for each ADC
- Two link interfaces, that serialize a 16 bit quantity according to the TRAM link protocol
- An interface to the CCDC bus
- An interface to the RAM memory, with all the necessary logic to implement two separate FIFO buffers
- A state machine for the overall control.

The chip operates with a basic cycle time of 50 ns. On each cycle, it is possible to transfer data between two subunits. Each subunit manteins one "ready" flag, that signals whether it is capable of generating/sinking a data word. Once transferred, the word is processed autonomously by the receiving subunit.

Only a limited set of transactions are possible:

- From the memory to one of the two link interfaces
- From one of the two input registers to the memory
- From the bus interface to the memory, or viceversa (to store/retrieve protocol words)
- From the FIFO controller to the bus interface, or viceversa (to set/check status)

Therefore at most 8 different transactions are possible. If all are required, a cycle time of 400 ns results.

The state machine decides what transaction to perform next according to some scheduling algorithm. The simpler one cycles through all 8 states, skipping the transaction if the two subunits involved are not both ready.

Since usually most of the transactions are mutually exclusive, a different scheduling, that chooses dynamically or according to priorities which transaction to perform, results in a much faster cycle time. For example, since all transactions listed above except the first involves the 56000 DSP and are therefore mutually exclusive, the algorithm could cycle between only four states, two for each channel. For each channel it would transfer, if possible, a word from the FIFO to the TRAM link, and then it would perform one of the other three transactions, choosing with a priority criterion.

For the first version of the controller, we have chosen one of the simplest algorithms, that cycles through 8 states. It is described in section 2.5.

The top level schematic for the FPGA is shown in fig. 4. The structure organization in submodules is clearly visible, all modules have typically a few `RDY` and `STB` lines beside the data bus.

The clock is common to all modules, the chip operates in a totally synchronous mode. Two clocks are used. The main clock operates at 20 MHz, with a single clock cycle for each state. An auxiliary clock at 40 MHz is used to syncronize some fast input signals.

2.1 Input register

The input register (fig. 5) is simply a 16 bit data register, that stores the results of the ADC conversion. It is strobed by the EOC line in the ADC, and, after the data has been stored, the RDY line is raised. The data is placed on the bus and RDY is cleared when the READ strobe is asserted.

The transition on the `EOC` line is sensed with a syncronous logic. The signal is active low, and therefore is inverted at the input.

According to the data sheet, the converter presents valid data at its output on the leading (falling) edge of `EOC`. However, it often generates glitches both on the data lines and on `EOC`, causing false data to be sampled. Therefore the input logic requires that `EOC` is held valid for at least two consecutive clock periods (50–100 ns) to be acknowledged. This is marginal for the fastest converters (4022), that are however not used in NICS.

2.2 TRAM link register

This submodule serializes the data according to the TRAM protocol. When the output link is in the ready state, the RDY line is asserted. When the WRITE line is asserted, a 16 bit data word is read from the bus, and the RDY line is deasserted. The data is then placed on the link as two bytes, waiting for the acknowledge from the transputer serving that link.

The submodule contains a one-byte buffer to store the outgoing byte while the link register is reloaded.

The submodule is composed of three main units. A register (fig. 6) stores the incoming word, and can send one of the two bytes to a shiftregister, using a 2-1 multiplexer (fig. 7). Shiftregister and multiplexer are controlled by a state machine (link ouput sequencer, fig. 8).

When a data is written to the register and the `WRITE` line asserted, the output sequencer is started. If the READY flip flop is set, it first sends two mark bits on the output line, while the shiftregister is loaded. and resets the `READY` flag. The data bits are shifted out, followed by at least one stop bit. The process is repeated for the second byte.

The input link is monitored using the fast clock (40 MHz), and any rising transition (0 to 1) is interpreted as an acknowledge bit. The acknowledge sets the `READY` flip flop, allowing subsequent bytes to be transmitted. No other signal is ever transmitted on this link, and thus a more complex logic to distinguish between acknowledge (single positive bit) and data (with two trailing mark bits) has not been implemented.

When the acknowledge for both bytes has been received, RDY is reasserted. This can happen before the actual data has been sent, allowing the register to be reloaded, and thus to operate the link at full speed.

The sequencer is asyncronously reset by the control line `RST_LNK`, from the control register.

2.3 FIFO memory

This section (fig. 9) interfaces the chip to the FIFO external RAM. It contains two set of address registers, one for each link. The set used is selected by the input line SEL. Each WRITE operation writes a 16-bit word in a successive memory location, in a cyclic (modulo $2^n - 1$) buffer. Each READ operation reads a successive word from the bottom end of the same buffer. The buffer status is signaled by the `FULL` and `EMPTY` output lines. The buffer is enabled by the control line `CE_D`.

The FIFO contains a control/status register (fig. 10, selected asserting the `CE_C` line. The status register allows the FIFO counters to be asyncronously cleared (buffer flush), and to drive some control lines. The status register allows monitoring the FULL/EMPTY lines and an external ID switch to be monitored.

The FIFO operates in a pipelined mode. When a word is written to the external memory, it is first transferred to a latch, while the associated address is computed. In the following cycle, the word is actually transferred to the memory. When a word is requested form memory, the operation requires three cycles. In the first cycle, the memory address is computed and latched. In the second cycle, the word is transferred from the memory to a storage register. In the third cycle, the word is transferred via the internal bus to the destination submodule.

The memory interface is strongly pipelined, to guarantee that the signal has sufficient time to propagate. In a memory write operation, the word is first written to the memory output register, together with the corresponding address, and in the *subsequent* cycle it is written to memory. A read operation requires three consecutive cycles. First, the address is generated and placed in the address register. In the subsequent cycle, data is read from memory and placed in an input register. In the third cycle, the word is transferred to its destination.

For homogeneity, a read operation requires three cycles even when the status register is addressed. In this case, the register content is placed on the bus on the third cycle.

Read and write cycles are placed on alternate cycles, in order to optimize memory bus utilization. In the following table an (hypothetical) example is shown in table 1.

Cycle no.	Internal bus	memory bus
1	Word 1, Input to MemReg	-
2	Addr. generation for word 4	Word 1, Xilinx to Mem
3	Word 2, Input to Memory	Word 4, Mem to Xilinx
4	Word 4, MemReg to output	Word 2, Xilinx to Mem

Table 1: Example of pipelined memory access

The external memory is addressed as two banks of 64Kword. The most significant bit of the address, ADDP16, selects the bank. Inside each bank, addresses are generated with a pseudo-random maximal length counter (fig. 11). This kind of counter generates addresses in a sequence of length $2^n - 1$. It is quite simple to compare two addresses for consecutiveness, since consecutive addresses are shifted by one bit. Each bank has two counters, respectively for the top and bottom of the queue. When addresses are equal, the queue is empty. When the bottom address is the successive of the top address, the queue is full. Full and empty status are available as `RDY` lines, in the FIFO status register, and as external lines used to drive four LEDs.

The FIFO is in the ready status if it is not empty. If a queue reaches the FULL status, and another push operation is attempted, the operation is not performed and an error flag is set. The two `FULL` flags are available to the DSP on the control status register, and drive an external LED on the front panel. This flag must be explicitly reset by the DSP.

The control register is 8 bit wide, occupying only the 8 least significant bits of the DSP word. In a read operation, the unused bits are undefined. During write, four control lines, as described

in table 8, are set. RST0 and RST1 reset the FIFO buffers. RST_LNK resets the output links (they are left in the READY state). RST_FULL resets the two FULL flags and LEDs.

The status register reports a board ID in bits 4–7, and the status of the EMPTY lines and FULL flags. These two last are latched, since a full status, even momentary, disrupt link syncronization, and is thus an irrecoverable error.

2.4 DSP Bus interface

The Xilinx chip is interfaced to the CCDC bus of the 56001 DSP. The DSP can write any word in the FIFO memory, to implement complex OCCAM protocols, and can write and read the FIFO status register.

The DSP interface logic is described in fig. 12. The READ and WRITE lines are syncronously sampled at 40 MHz, to avoid glitches while allowing detection of the short DSP strobes.

When the DSP writes to the FPGA, the data is latched in an internal register, together with the A1 and A0 bits of the address. The RDY_R line is then asserted, to signal that a word is ready to be read. The data is placed on the bus when the STB_R line is asserted.

When the DSP reads the FPGA, the RDY_W line is asserted. Bits A0 and A1 are latched to select the correct data source. Data from the internal bus is transferred on the DSP bus when STB_R is asserted, and latched internally on its falling edge. The bus cycle can then be completed. The DSP must introduce enough wait states to guarantee that the DSP bus cycle is completed after the sequencer has placed the data on the bus. With the sequencer described there, the worst case delay is 12 cycles, corresponding to 600 ns.

The DSP bus provides also the main system reset signal. It resets all the internal logic, including FIFO buffers and output links. This signal is also available negated on an output pin for the TRAM modules.

2.5 Sequencer

The sequencer (fig. 13) analyzes all the status lines from the remaining sections, and generates all the control lines. In the first version, it follows a cyclic schedule algorithm (slow but simple to implement). It is composed of a modulo-8 counter, and an appropriate combinatorial logic that generates the STROBE signals depending on the current time slot (phase) and the status (READY) lines.

The phases are listed in tab. 2.

Phase	Bus operation	Memory operation
0	Idle	Idle
1	DSP-bus read	
2	DSP-bus write	Fetch word for DSP
3	Send word to DSP	Store DSP word
4	Read ADC 0	Fetch word for link 0
5	Write link 0	Store ADC 0
6	Read ADC 1	Fetch word for link 1
7	Write link 1	Store ADC 1

Table 2: Sequencer cycles

All RDY lines are sampled in the phase preceding the one in which the corresponding subunit is used. For example, the input register 0 ready flag is sampled during phase 3, since the register is read during phase 4. Actually, the phase counter is one cycle ahead with respect to the sequencer output, and all strobe signals are latched and delayed one clock cycle.

For FIFO read operations the sequencer must generate both the appropriate commands to the FIFO module and, two cycles later, the commands for the receiving module. For example, when link 0 is ready, the FIFO is addressed for read in cycle 3, and the link 0 module is addressed in cycle 5 to store the generated word.

The maximum instantaneous conversion rate is equal to the total cycle time, i.e. $8 \times 50\text{ns} = 400\text{ns}$. Since conversion is not performed syncronously, extra time must be allowed for worst case

timing. The worst case occurs when the EOC line is strobed at the beginning of phase 0 (for ADC 0). The RDY9 line is raised with a maximum delay of 4 clock cycles, at the beginning of phase 4. The line is sampled during phase 3, and the register is read into the FIFO at the next phase 4, after 12 cycles. In this case, the register can be reloaded after 13 cycles (650 ns), for a maximum conversion rate of 1.5 MHz.

This value could be reduced by optimizing the phase scheme, eliminating phase 0 and overlapping phase 1 and 7, and using a faster logic for the EOC detection logic.

Sequencer operation depends on the overall device status, summarized by the ready lines, that are grouped in a bus for simplicity. The correspondence of these signals to the submodules lines is listed in table 3.

Line	Submodule	line	Asserted when
RDY0	DSP	RDY_R	Data available from the DSP bus
RDY1	DSP	RDY_W	Data required for the DSP bus
RDY2	FIFO	READY0	Data available on FIFO bank 0
RDY3	FIFO	FULL0	Buffer full on FIFO bank 0
RDY4	FIFO	READY1	Data available on FIFO bank 1
RDY5	FIFO	FULL1	Buffer full on FIFO bank 1
RDY6	Link0	READY	Link 0 available for sending data
RDY7	Link1	READY	Link 1 available for sending data
RDY8	ADC0	READY	Data ready on ADC 0
RDY9	ADC1	READY	Data ready on ADC 1
RDY10	DSP	C/D	Selects data(0) or control reg.(1)
RDY11	DSP	SEL	Selects FIFO bank

Table 3: READY lines

Combining these signals and the value in the phase counter, the sequencer generates the appropriate strobe signals. This is done in a combinatorial array, that generates 10 strobe signals, STB[9:0]. After the combinatorial logic, a latch has been added. In this way, all the strobe signals are clean, without spurious glitches. The drawback is a 50 ns latence added for the response to any external signals. Moreover, the strobes for the link write and DSP bus write are delayed two cycles with respect to the other signals, to account for the time needed by the FIFO memory to retrieve the data word.

The correspondence between the STROBE signals STB[9:0] and the input of the chip submodules is listed in tab. 4.

Line	Submodule	line	Asserted for
STB0	DSP	RDY_R	Data read from the DSP bus
STB1	DSP	RDY_W	Data write for the DSP bus
STB2	FIFO	CE_D	Data read/write on FIFO buffer
STB3	FIFO	CE_C	Data read/write on FIFO control reg.
STB4	FIFO	SEL	Select FIFO bank 0/1
STB5	FIFO	R/W	Read (1) or Write (0) operation
STB6	Link0	READY	Write on link 0
STB7	Link1	READY	Write on link 1
STB8	ADC0	READY	Data read from ADC 0
STB9	ADC1	READY	Data read from ADC 1

Table 4: STROBE lines

3 Board implementation

The ADC board is implemented in a 6-layer PCB. It has only minor modifications with respect to the previous one (without FIFO). It hosts two ADC modules, two TRAM modules, and a Xilinx

chip. To better insulate the analog and digital parts, the topmost layer of the PCB is a ground plane, and the Xilink chip and memories are mounted as surface mount components on the bottom side.

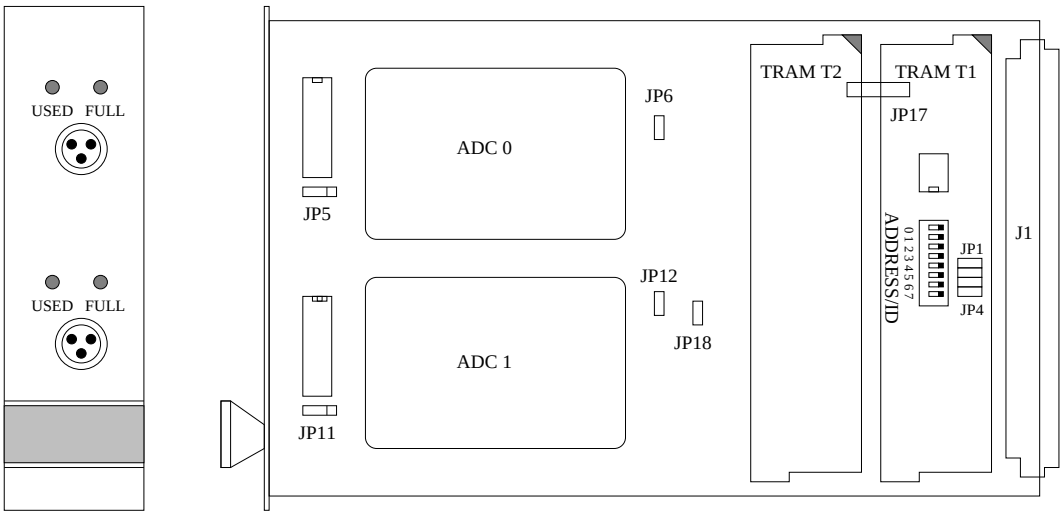

Figure 2: Board layout and jumper location on the ADC FIFO board

Power supply for each ADC is derived independently from a $+/-18$V supply on the CCD bus using separate regulator IC's.

To accomodate all the required signals, the Xilinx chip uses a 160-pin package. It can directly drive the DSP bus without external buffers.

3.1 Analog section and gain setting

The analog section of each ADC channel is composed of a fast instrumentation amplifier (BB OPA110), and a fast ADC converter. The gain of both the amplifier and the ADC can be preset in hardware. The amplifier has a factory preset gain of 1. To obtain higher gain, an external resistor is needed (R7 for channel 0, R9 for channel 1). The resistor value is given by the formula $R = 6K\Omega/(G-1)$, where G is the desired gain. For example, a gain of 10 is obtained with a resistor of 667Ω.

The ADC gain setting is done with solder jumpers, located on the bottom side of the board. Jumper settings for the available ADC ranges are listed in tab. 5. A X denotes a jumper soldered close, a - a jumper left open.

Ch. 0 Ch. 0	JP7 JP13	JP8 JP14	JP9 JP15	JP10 JP16
$+/-2.5$V	X	-	X	-
$+/-5.0$V	X	-	-	X
$+/-10.$V	-	X	-	X

Table 5: Gain setting for ADC

The input of the ADC chips can be shorted to ground, using jumpers JP5 and JP11, resp. for channel 0 and 1. Analog and digital grounds in each board can be left unconnected, or can be connected with solder jumpers JP6-JP12.

The ADC start conversion lines can be selected from CCDC bus lines STA12-15, using jumpers JP1-JP4 (see tab. 6). ADC0 can be driven by STA12 (JP1) or STA14 (JP3). ADC1 can be driven by STA13 (JP2) or STA15 (JP3). Usually in a 4 channel system one card has JP1 and JP2 selected, and the other has JP3 and JP4.

	JP1	JP2	JP3	JP4
Ch. 0:	STA12	STA14	-	-
Ch. 1:	-	-	STA13	STA15

Table 6: Start of conversion line selection

3.2 TRAM link configuration

The Xilinx chip receives the EOC and data lines from the ADC. The ADC produces unsigned words, from 0 (maximum negative value) to $2^{16} - 1$ (max. positive value). Since transputers deal only with signed integers, the MSB of the word from the ADC is inverted in hardware in the input register of the Xilinx.

The Xilinx generates two TRAM links, routed to link 0 of two TRAM slots. Link 1 and 2 of each TRAM slot are routed respectively to the preceding and following cards in the DSP bus. Link 3 interconnects the two slots. This connection is depicted in fig. 3.

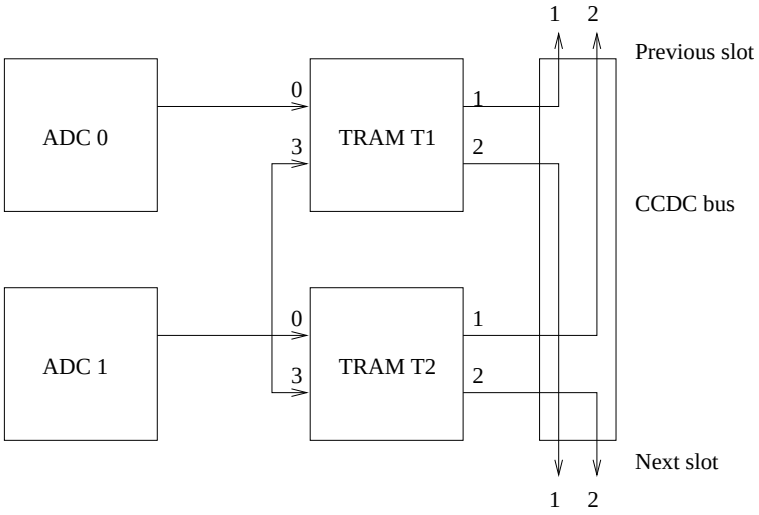

Figure 3: ADC board link configuration

Usually, one or two TRAM modules are placed in the two slots T1 and T2. It is possible to jumper in various modes the unoccupied slot(s), to route the links from the Xilinx chip to appropriate transputers. Possible configurations include:

- No TRAM modules. Links 0 of each module are routed to link 1, to a TRAM module in a previous board in the DSP bus.
- One TRAM module in slot T1. Slot T2 has link 0 connected to link 3, and the transputer on T1 reads both ADC's. A jumper between link 1 and 2 on T2 is required to propagate the second link chain. This si the normal configuration for the NICS controller.
- Two TRAM modules, each one reading one ADC.

3.3 Xilinx programming

The Xilinx chip can be programmed either using a serial PROM, or by an external programmer (Xchecker) connected to a PC serial port. Selection of the programming source is done using jumper JP18. If this jumper is open, the chip is programmed by a Xchecker probe, connected to JP17. Signals on JP17 are listed in table 7. It is possible only to program/reset the chip, and not to read back its status.

If JP18 is shorted, the chip is programmed using a serial PROM, U11. To avoid electric contention, the PROM must be removed frm the socket while using the Xchecker probe, and viceversa.

Cycle no.	Internal bus	memory bus
1	VCC	Power supply for the probe
2	GND	
3	n.c.	Mechanical key
4	CCLK	Programmer clock (in)
5	DONE	DONE signal (out)
6	DIN	Data input (in)
7	PROG	Program command (in)
8	INIT	Initialization in progress (in/out)
9	RESET	Chip reset (in)

Table 7: Xchecker probe connector pinout

3.4 Address space and control register

The board is addressed on the DSP as a 16-word region in the 256-word page occupied by the CCD controller. Only the 3 first words are used, resp. to push/pop a word in the two FIFO's, and for the control register. Successive words are shadow copies of these first three.

The region in the address space of the DSP is selected using a DIP a DIP switch. Bits 4–7 of the switch correspond to the same bits in the DSP address space, thus selecting one of 16 possible positions in the 256 word CCDBUS decoded space. The other 4 bits are used for board identification, and are read back using the status register. The DSP can implement arbitrary protocols on the TRAM links, by writing appropriate preamble words on the two FIFO buffers. Since the DSP controls also the ADC strobe lines STA12-15, the sequence of data stored in the buffer depends uniquely on the DSP program flow.

The DSP can also read the FIFO. This is useful for debug purposes only, since the FIFO buffers are emptied by the transputers in a completely unpredictable mode. Therefore if data are fetched from the buffer when transputers are present, data order on the TRAM links is disrupted and most likely the transputer program will hang.

The control word is used only to reset the FIFO address counters. Bit 4 of the word written in the control register resets counters for FIFO 0, and bit 5 resets those for FIFO 1. Bit 6 clears the output links, interrupting any activity and placing them in the `READY` state. Bit 7 resets the two `FULL` flags in the control register.

The status word is a 8-bit register. The 16 MSB of the DSP word are unused, and must be masked off in the DSP program. Bits 0-7 are listed in tab. 8. The `FULL` bits are latched, and must be explicitly cleared. The `EMPTY` bits reflect the instantaneous state of the FIFOs.

When accessing the chip for reading, the DSP must specify at least 10 wait states, for a total time of 600 ns, to accomodate for the worst case delay between beginning of the read operation and the availability of data on the bus.

Bit	Status register
0	Empty 0
1	Full 0
2	Empty 1
3	Full 1
4-7	ID switch

Bit	Control register
0-3	Unused
4	Reset FIFO 0
5	Reset FIFO 1
6	Reset output links
7	Reset FULL flags

Table 8: Status and control registers

References

[1] G. Comoretto, C. Baffa, F. Lisi: "NICS–Camera infrarossa TNG: Descrizione generale dell' elettronica di controllo e presa dati", Osservatorio di Arcetri, rapporto tecnico n. 4/95

[2] C. Baffa, G. Comoretto: "Il software per la camera infrarossa NICS", Osservatorio di Arcetri, rapporto tecnico n. 3/96

[3] Bortoletto F. et al. (1992): "CCD Sequencer User Manual", rapp. int. Osservatorio di Padova

[4] : G. Comoretto, C. Baffa: "NICS - Galileo Near Infrared Camera. Programma di controllo per il DSP sequencer", Osservatorio di Arcetri, rapporto tecnico n. 4/98

[5] Lisi et al. (1994): "The IR Imager/Spectrograph", in Fusi Pecci F., Stirpe G. Zitelli V.: "TNG Instrument Plan: II. A progress report", Osservatorio Astronomico di Bologna, WWW: `http://www.bo.astro.it/galileo/galbo.html`

[6] Inmos ltd (1989): "IMS T800 Transputer Engineering Data", ordering code 42-1406-02

List of Figures

List of Tables

Contents

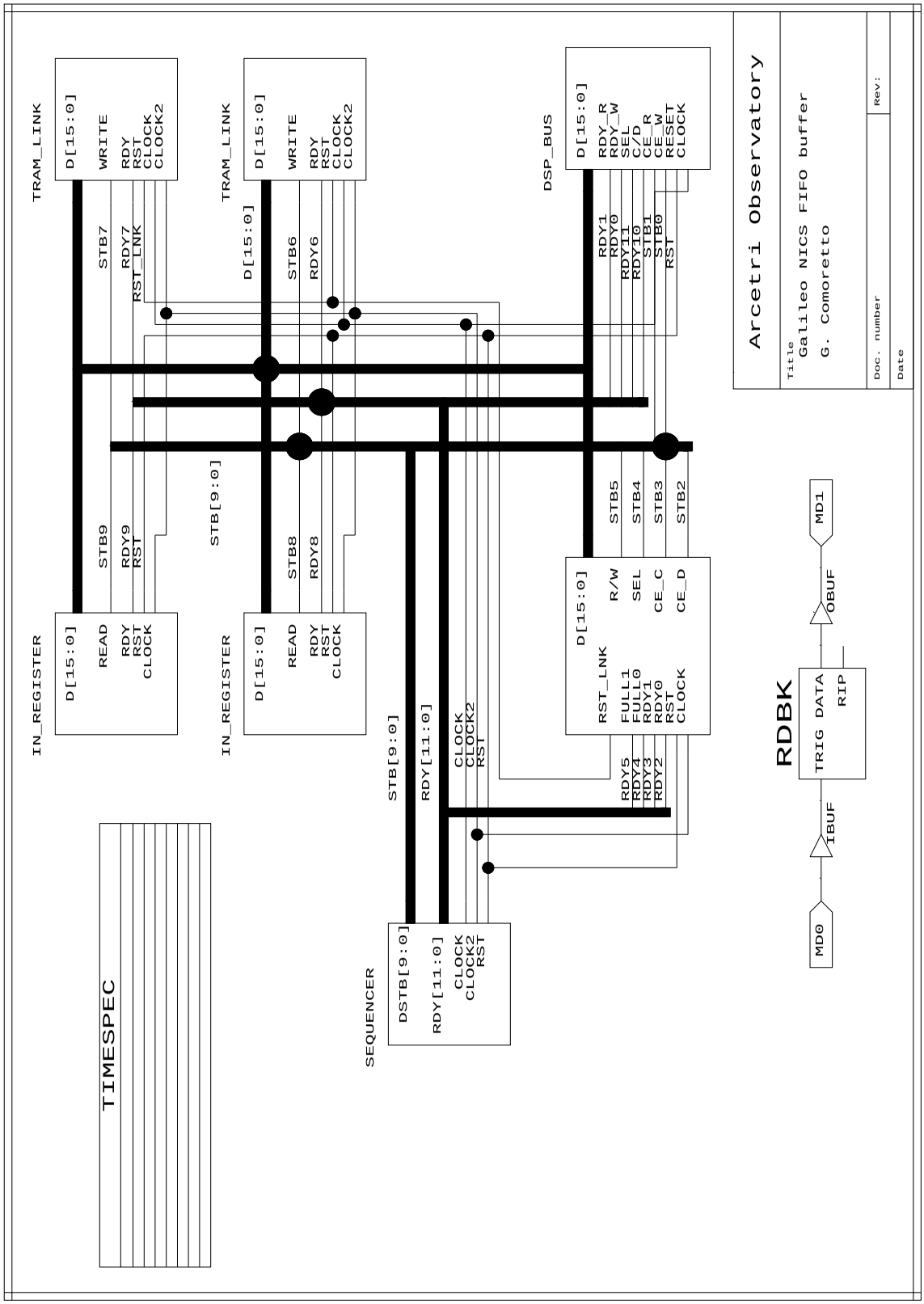

Figure 4: Global architecture of the FIFO chip

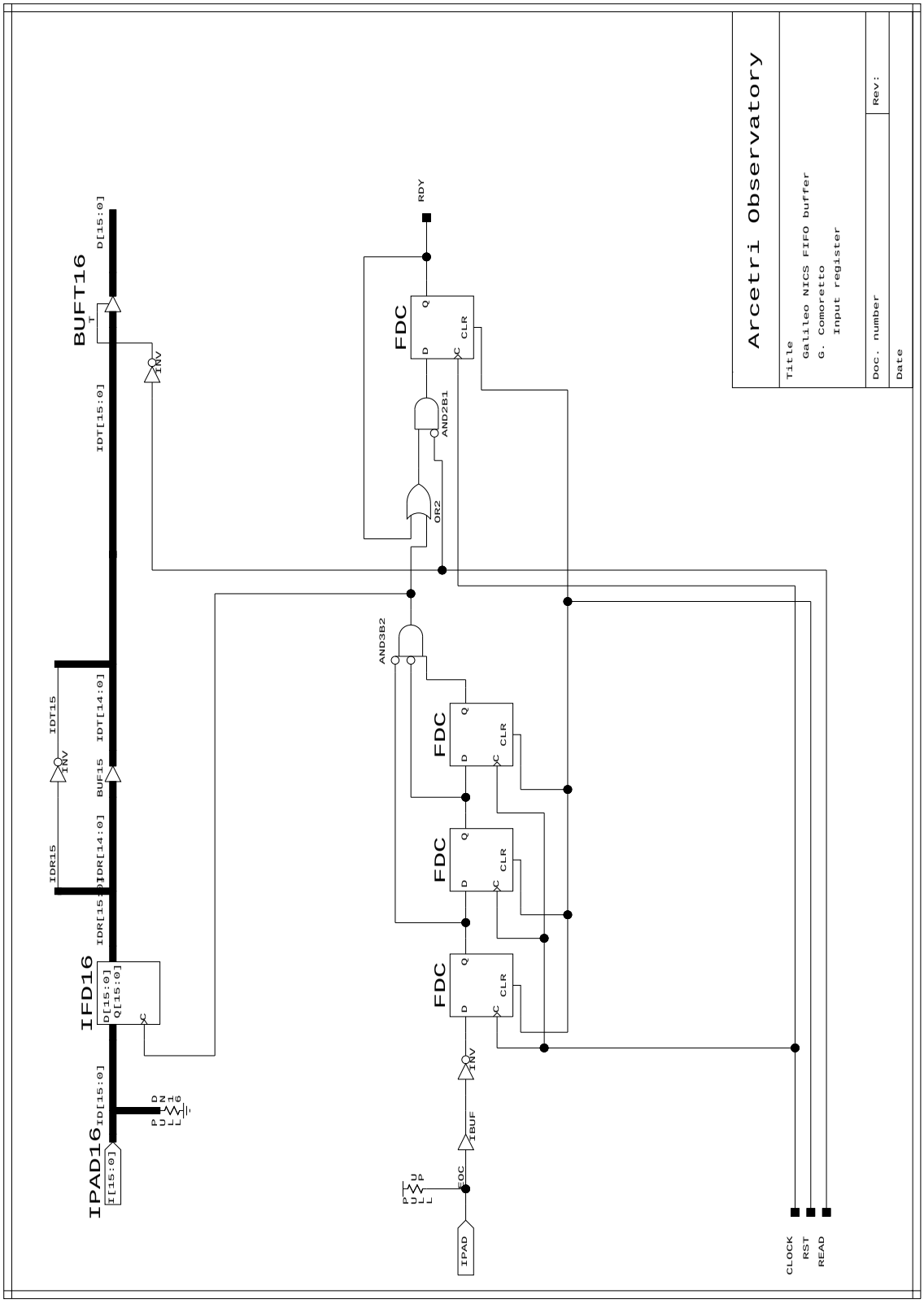

Figure 5: Input register

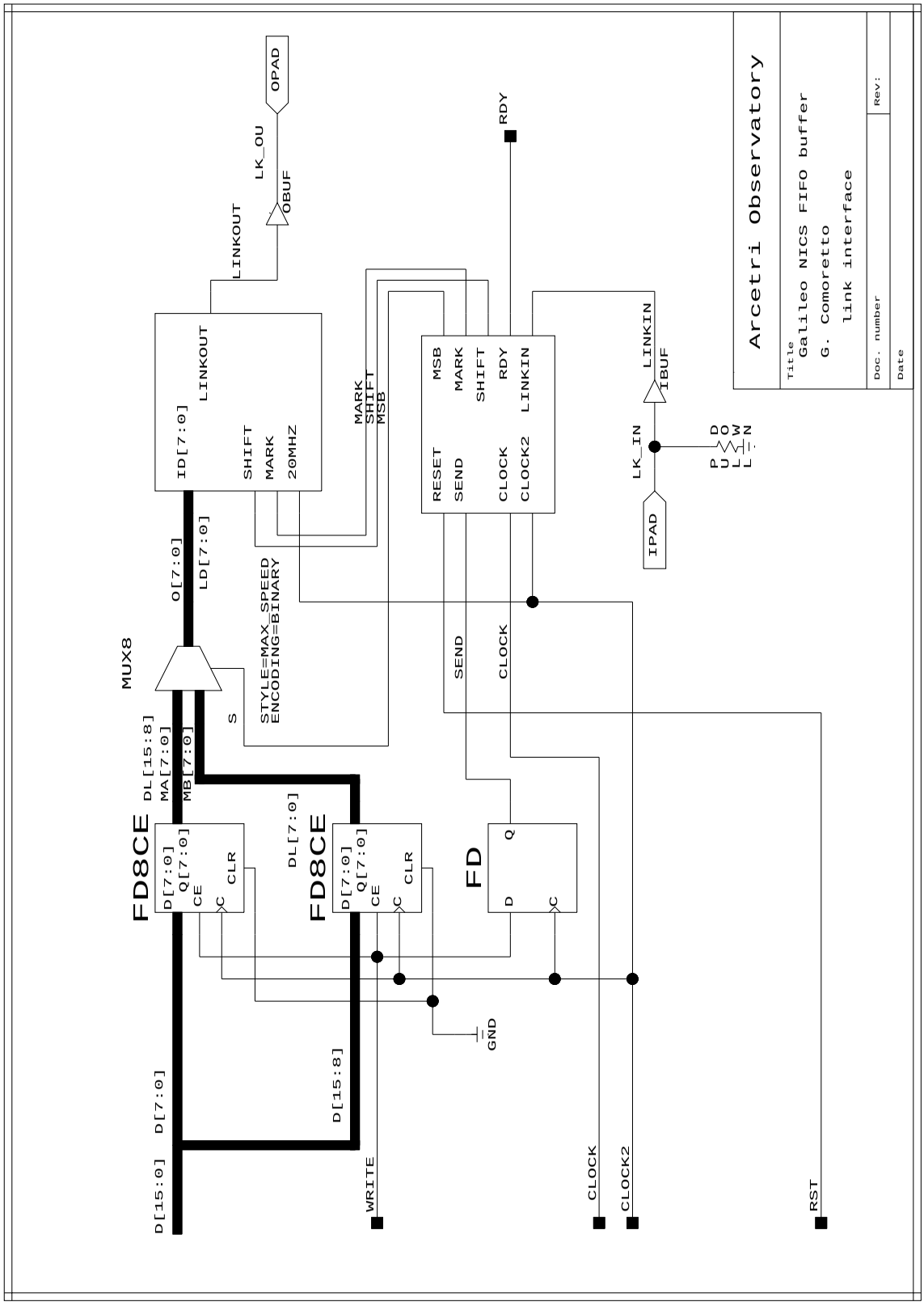

Figure 6: TRAM link register and multiplexer

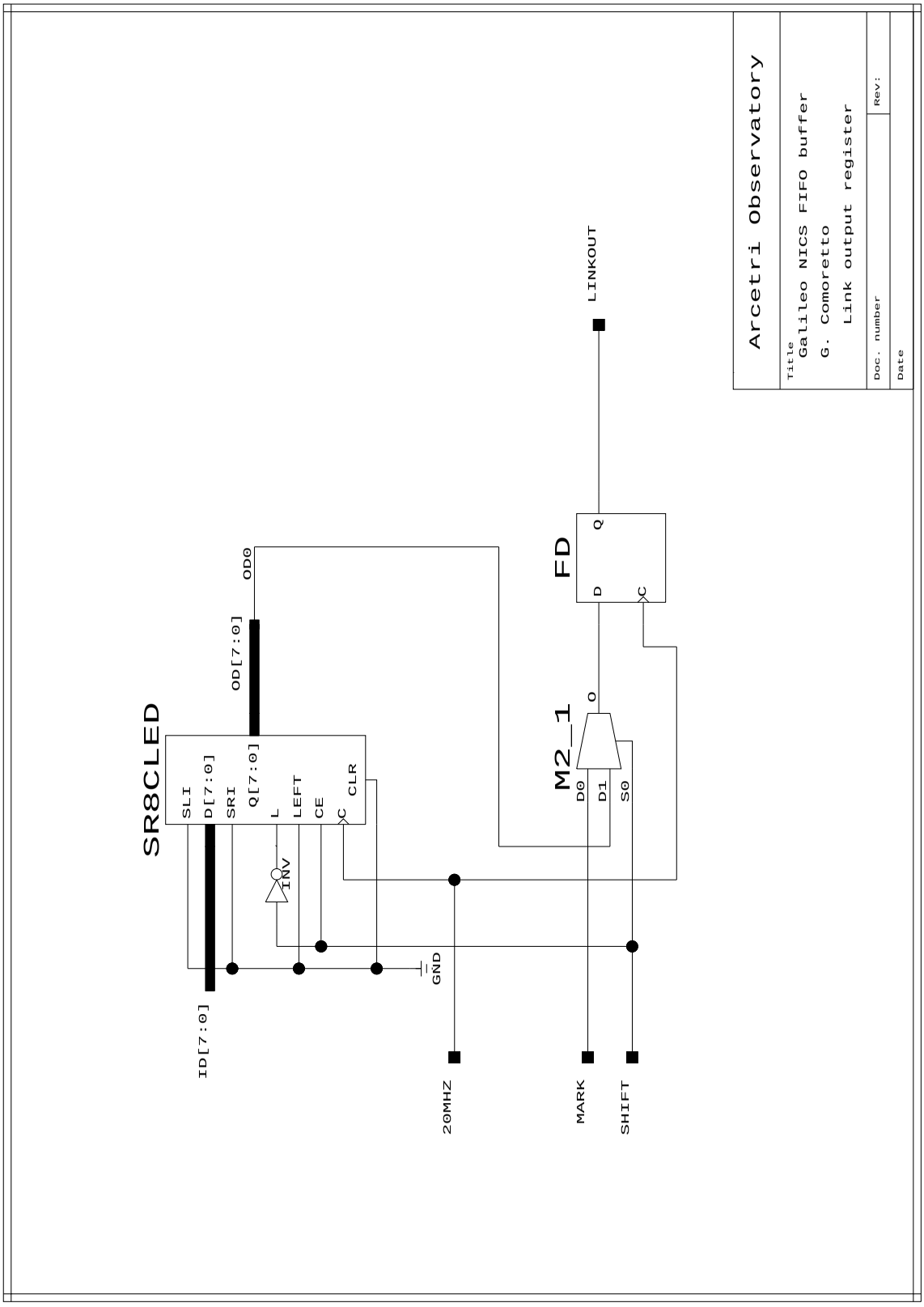

Figure 7: TRAM link output shiftregister

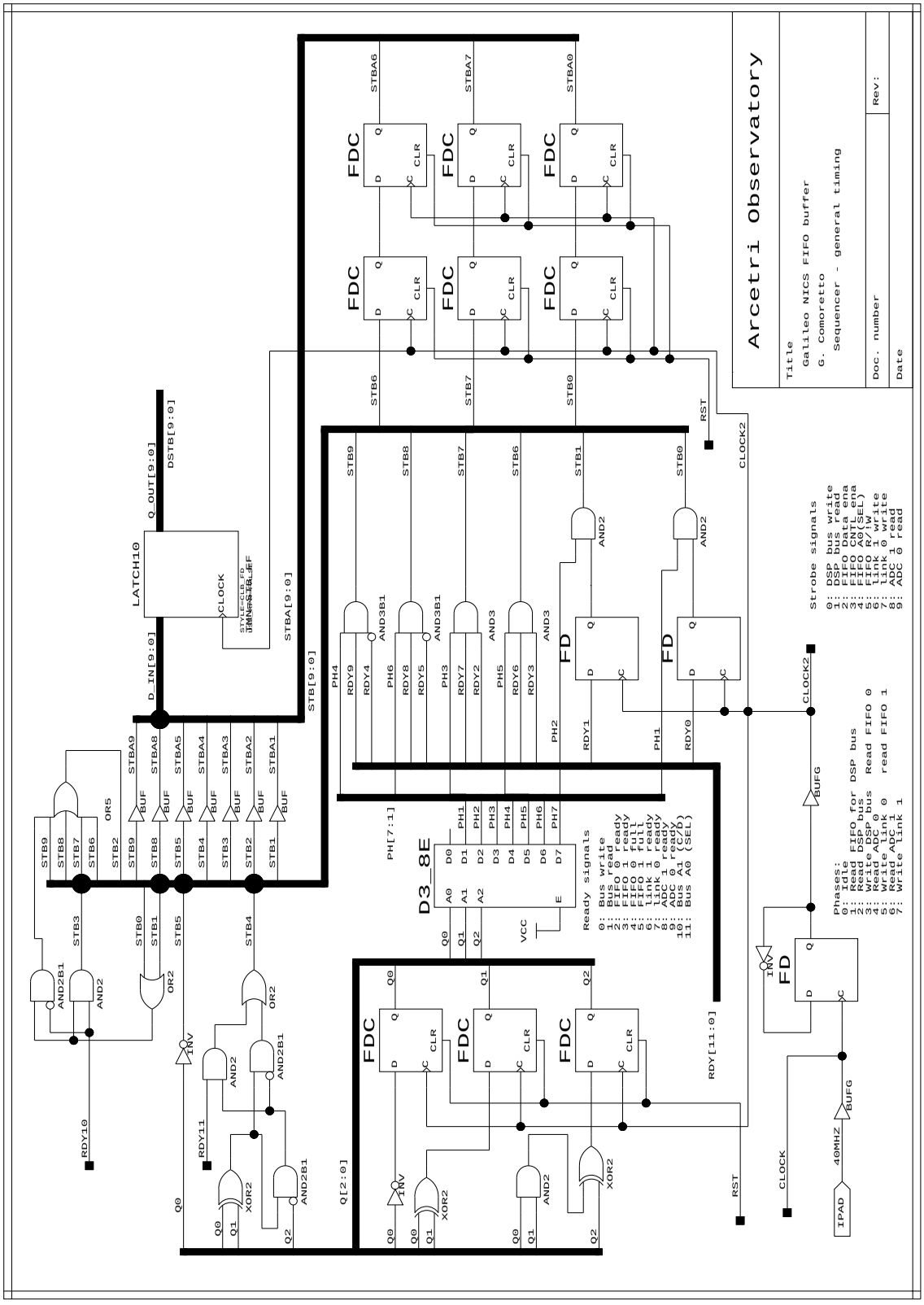

Figure 8: TRAM link sequencer and READY logic

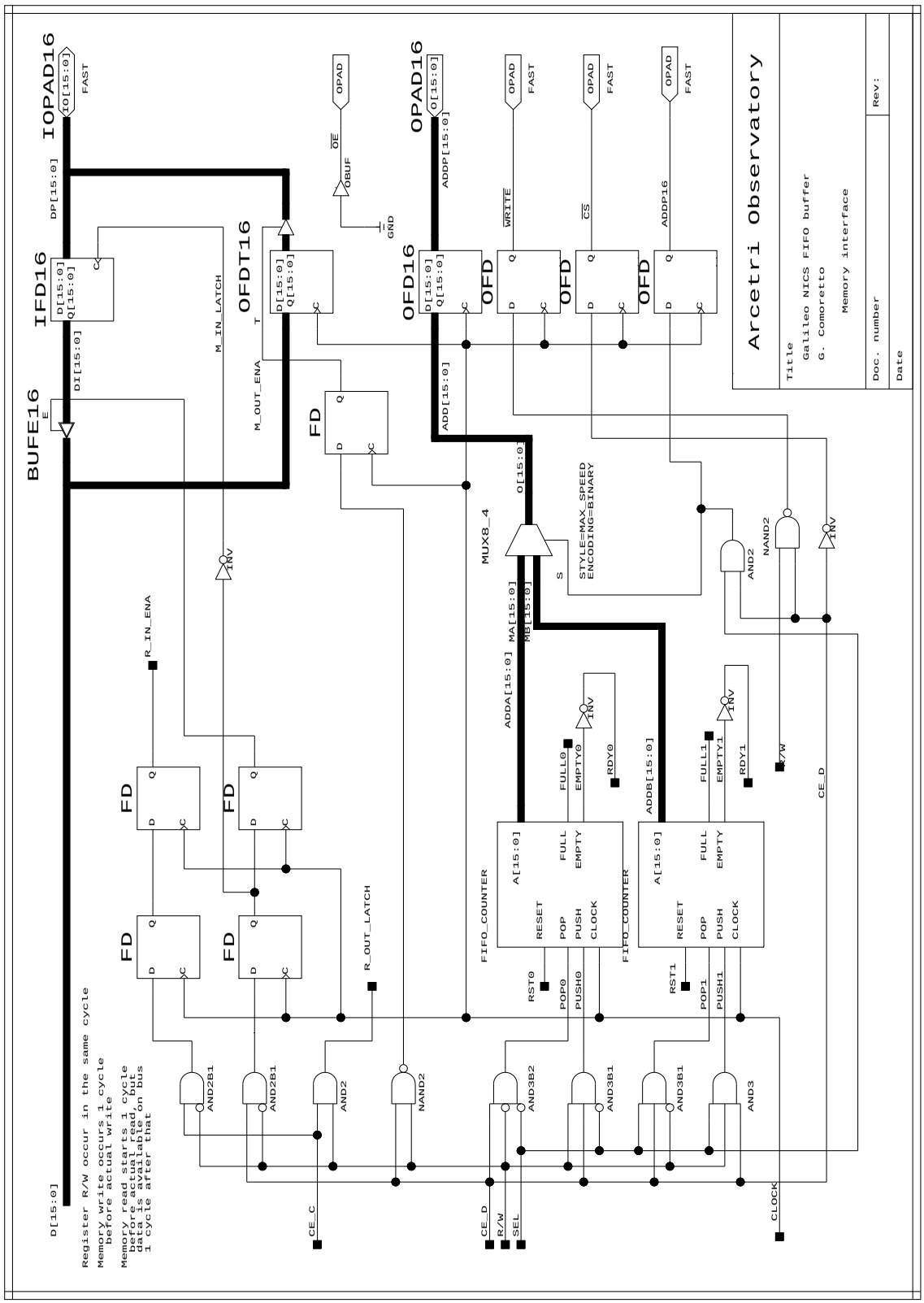

Figure 9: FIFO memory controller

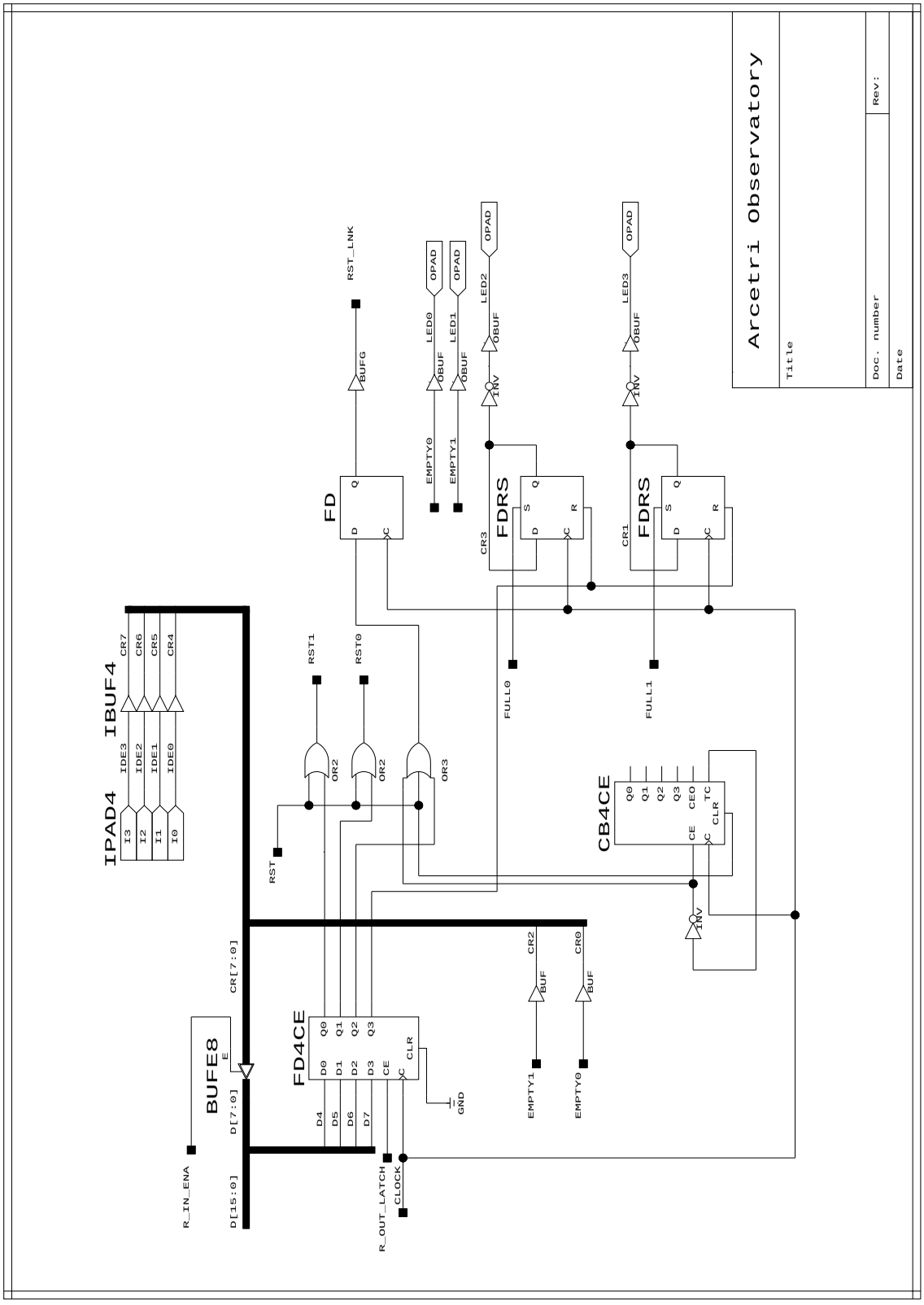

Figure 10: FIFO memory: control and status register

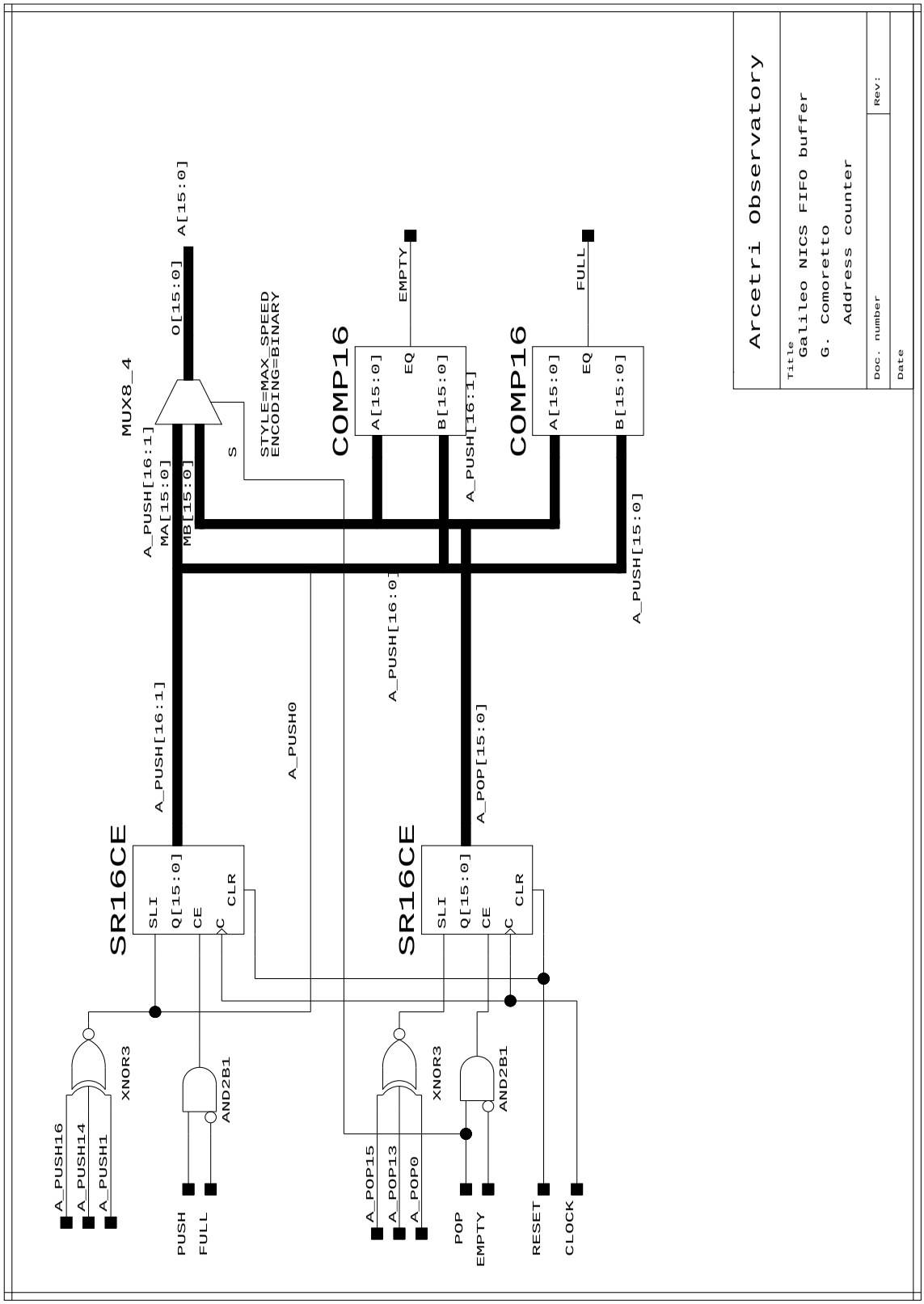

Figure 11: FIFO memory: address counter

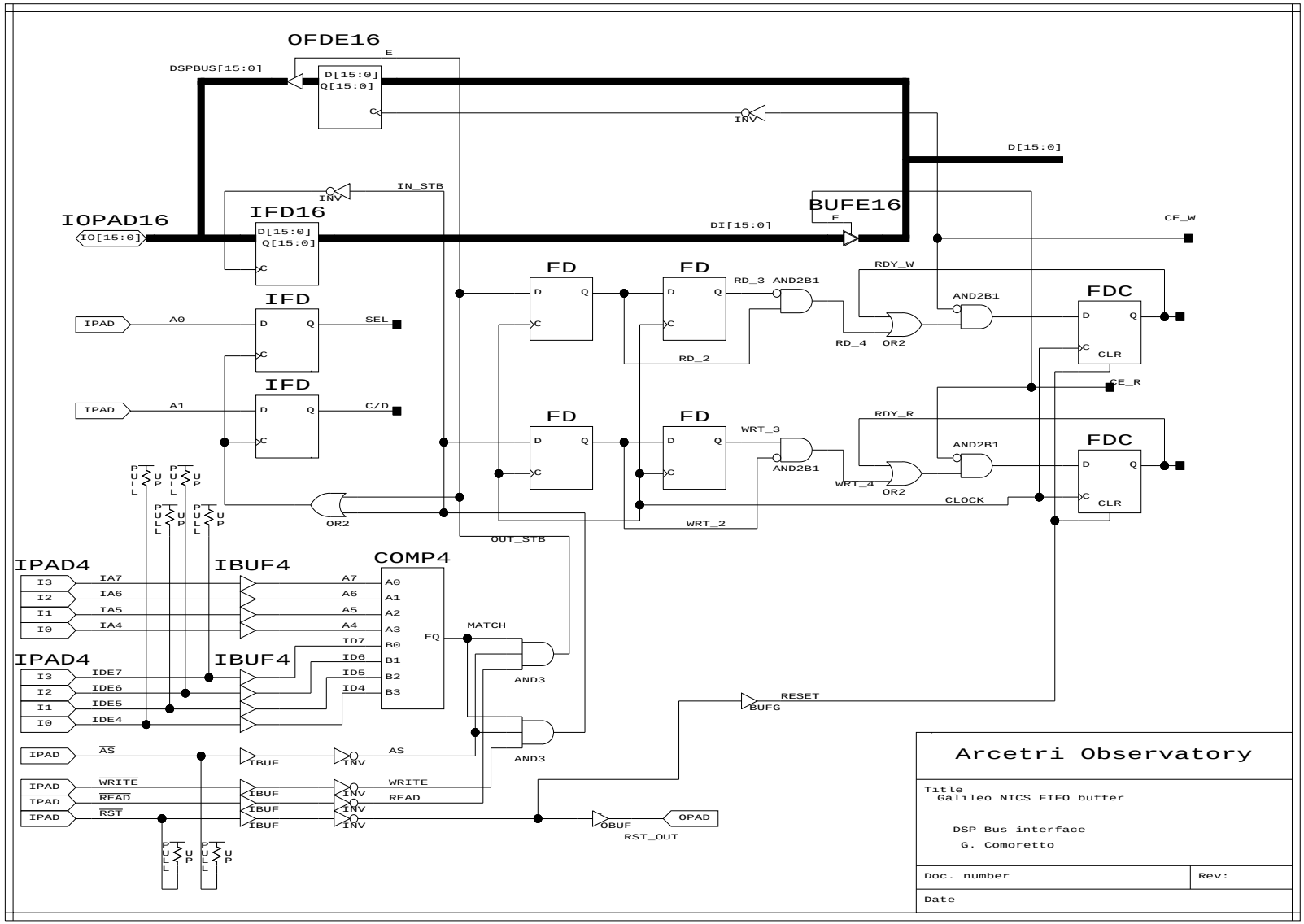

Figure 12: DSP bus interface

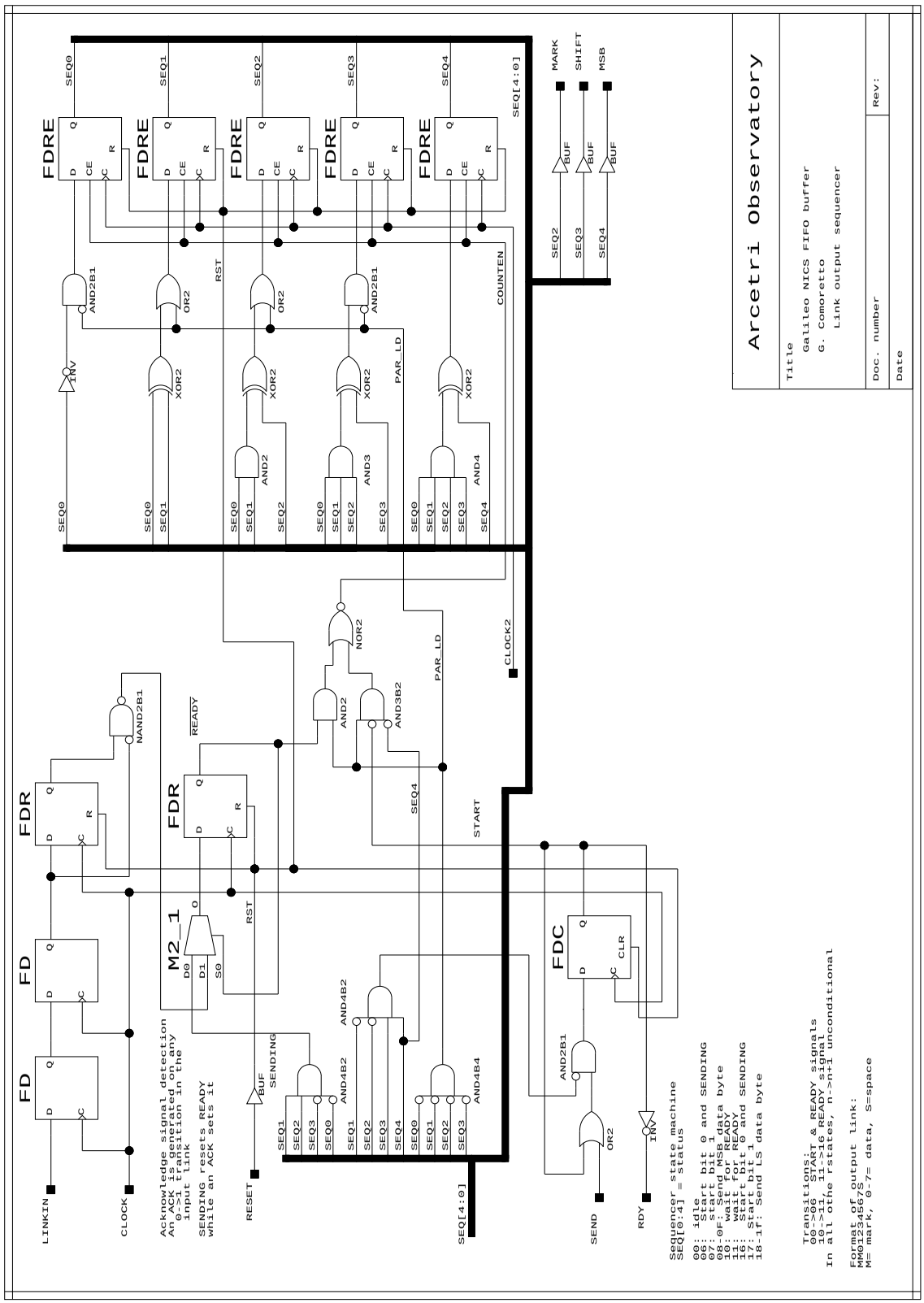

Figure 13: Sequencer and timing logic